

# Simulating hard rigid bodies.

C. De Michele\*

*Dipartimento di Fisica, Università di Roma "La Sapienza",  
P.le Aldo Moro 2, 00185 Roma, Italy*

---

## Abstract

Several physical systems in condensed matter have been modeled approximating their constituent particles as hard objects. The hard spheres model has been indeed one of the cornerstones of the computational and theoretical description in condensed matter. The next level of description is to consider particles as rigid objects of generic shape, which would enrich the possible phenomenology enormously. This kind of modeling will prove to be interesting in all those situations in which steric effects play a relevant role. These include biology, soft matter, granular materials and molecular systems. With a view to developing a general recipe for event-driven Molecular Dynamics simulations of hard rigid bodies, two algorithms for calculating the distance between two convex hard rigid bodies and the contact time of two colliding hard rigid bodies solving a non-linear set of equations will be described. Building on these two methods, an event-driven molecular dynamics algorithm for simulating systems of convex hard rigid bodies will be developed and illustrated in details. In order to optimize the collision detection between very elongated hard rigid bodies, a novel nearest-neighbor list method based on an oriented bounding box will be introduced and fully explained. Efficiency and performance of the new algorithm proposed will be extensively tested for uniaxial hard ellipsoids and superquadrics. Finally applications in various scientific fields will be reported and discussed.

*Key words:* Event-driven Molecular Dynamics, Molecular Liquids, Hard Rigid Bodies, Computer Simulations

*PACS:* 02.70.Ns, 83.10.Rs, 07.05.Tp

---

## 1. Introduction

Systems which are composed of many particles can often be modeled as an ensemble of hard rigid bodies. Such description is particularly successful when excluded volume interactions are dominant and internal vibrational degrees of freedom are negligible. Despite the absence of any attraction, particles interacting with only excluded volume

---

\*Tel.: +390649913524; fax: +39064463158.

*Email addresses:* `cristiano.demichele@roma1.infn.it` (C. De Michele )

interactions exhibit a rich phase diagram with a multiplicity of phases, especially when the shape is a non-spherical one[1, 2, 3, 4, 5].

Hard Spheres (HS) are a classical example of hard-body model, which has been particularly useful to understand the basic correlation which develops in simple fluids [6, 7, 8, 9, 10] and provides hints on the slow dynamics which characterize liquids approaching the glass transition, where packing effects become even more significant [11, 12, 13]. Even in the case of molecular fluids, HS models are a good starting point for sophisticated liquid matter theories [6].

Non-spherical models of rigid bodies are crucial to understand the role of the rotational degrees of freedom, as well as the role played by the shape in determining the system's physical properties. Onsager, introducing a hard spherocylinder model (HSC) for liquid crystals, showed that, by only changing the aspect ratio of the particles, a nematic phase can become thermodynamically stable [14]. In Onsager's theory, the internal energy of the system is zero and only the entropy, coming from translational and rotational degrees of freedom of the particles, contributes to the free energy of the system. Systems showing an "entropically driven" phase transition have been extensively studied over the last 60 years [15, 16]. The study of such transitions has been based on extensions of the original Onsager's theory [17, 18, 19, 20], and complemented by experiments (for a recent review see [16]).

Most of the information for hard-body systems has been calculated using numerical simulations [19, 21]. Several numerical techniques have been developed in the past to simulate particles interacting with only excluded volume interactions. The essence of these numerical algorithms involves the evaluation of the overlap between different objects or, equivalently, their geometrical distance. The first simulations of HS [22] were carried out by Alder and Wainwright in 1957, and they provided the first evidence of a crystal phase in the case of spherical hard particles (disks and spheres). In 1972 Vieillard-Baron [23] published a numerical investigation of the phase diagram of a two-dimensional hard-ellipsoids (HEs) fluid, introducing an overlap criterion for HEs suitable for a Monte-Carlo (MC) simulation. Building on the work of Vieillard-Baron, Frenkel *et al.* [21] investigated in 1984 the phase diagram of a tridimensional system of HEs, through MC simulations. Perram and Wertheim [24] introduced a simpler overlap criterion, which has been recently used by Donev *et al.* to perform molecular dynamics simulations of HE [25, 26, 27] and which has been also generalized to any couple of smooth convex shapes [28, 29, 30, 31]. Simulations of particles with more complex shapes have also been reported. For example in 1986 Stroobants *et al.* carried out MC simulations of a system of hard spherocylinders (HSC), i.e. molecules consisting of a hard cylindrical rod of length  $L$  and diameter  $D$ , capped at each end by hard hemispheres also of diameter  $D$ . These simulations are computationally less demanding [32] than the HE ones. More recently, MC simulations of hard-cylinders (HCY) have been performed [33] in order to look for a cubatic phase, which had been reported for cut hard spheres [34]. The oblate version of HSCs (i.e. the solid resulting from the intersection of two spheres) has been investigated by MC simulations under the name of UFO (the name comes from the shape of the particles) [35].

Molecular dynamics simulations of hard bodies are less common than the Monte Carlo ones, since the implementation of the overlap criterion between hard bodies must be complemented with an algorithm estimating the collision time between them. The evolution of the system in configuration space is propagated from one collision to the

next, giving rise to the so-called event-driven molecular dynamics (EDMD). In EDMD the predicted collision times between hard bodies are computed and stored into a time-ordered event calendar (as it was first done for HSs by Rapaport [36], although different techniques also exist in literature [37]). Such technique has been recently extended to Brownian Dynamics of HSs [38, 39].

A basic scheme for simulating hard non-spherical bodies is based on the standard MD algorithms, where at the end of each time step a check for possible overlaps is performed and the simulation is “rewinded” when an overlap occurs [40, 41]. This scheme is obviously inefficient. An overlap potential of two hard bodies  $A$  and  $B$  is a function  $F(A, B)$ , such that  $F < 0$  if  $A$  and  $B$  overlap. In general it is not easy (and not necessarily possible) to find an analytic form for the overlap potential of two rigid bodies of arbitrary convex shape, except for the aforementioned cases of UFO, HSC, HCY, HCS and HE. Overlap potentials for hard rigid convex bodies can be found in [3].

An EDMD algorithm for non-spherical objects employing an event-calendar has been recently proposed by Donev *et al.* [25] and tested on several systems of hard particles [26, 29, 30, 31]. Such algorithm [25, 26] requires the use of overlap potentials [24, 23], like in MC simulations. In the present paper a different route to provide a general algorithm to simulate hard particles will be presented.

If the surface of two hard rigid bodies (HRB) is smooth enough (i.e. first and second order derivatives are defined over the whole surface) a possible overlap potential is provided by the minimum distance between the two surfaces (defining the distance as negative when two rigid bodies overlap). In this paper we illustrate a method to calculate the distance between two generic convex HRBs based on a Newton-Raphson (NR) method. We will also illustrate a simple algorithm to efficiently evaluate a guess of the collision contact point and time. Starting from this guess true collision point and time can be calculated by solving a reduced system of equations, again through a Newton-Raphson method. It is worth noting that this algorithm can handle grazing collisions, i.e. collisions in which two rigid bodies touch slightly [26], without tuning the algorithm parameters significantly. Furthermore, in the case of HEs and superquadrics (SQs), we illustrate a new kind of neighbour list based on oriented bounding boxes that can also be easily generalized to more complex shapes. Like the algorithm proposed by Donev *et al.* [30, 31], our method can be easily generalized to arbitrary convex shapes (but also decorated with localized patchy interactions, see below) with similar efficiency.

The present algorithm has been already applied successfully to the simulation and to the study of various systems. It has been implemented in the study of structural and dynamical properties of uniaxial HE [42, 43]. With this code, adding the possibility of having localized patchy square-well interactions, it has been possible to study the statics and the dynamics of primitive models of Water [44] and Silica [45], as well as the irreversible gelation of a model inspired by stepwise polymerization of bifunctional diglycidyl-ether of bisphenol-A with pentafunctional diethylenetriamine [46, 47, 48]. More recently this code has also been generalized in order to simulate a coarse-grained model of biological systems and primitive models of proteins.

In Section 2 we introduce the general algorithm for the EDMD of rigid bodies. This requires discussing the HRB motion, the evaluation of the distance and the time of collision among HRBs and the optimized linked list method. In Section 3 we specialize the results of Section 2 to the specific case of HEs; in particular, we discuss a new nearest-neighbours list method, that over-performs the simple linked lists method usually

employed in EDMD in the case of very elongated HRBs. In Section 4 the performance in the case of HEs at various densities, elongations and with respect to the main parameters of EDMD is analyzed, while in Section 5 the performance of the algorithm in the case of SQs is investigated. In Section 6 some perspectives and applications of this new algorithm are discussed, and in Section 7 conclusions are drawn.

## 2. An event-driven algorithm for rigid bodies.

### 2.1. Geometry of rigid bodies.

The orientation of a HRB can be represented by the 3 column eigenvectors  $\mathbf{u}_i$  (with  $i = 1, 2, 3$ ) of the inertia tensor expressed in the laboratory reference system. These vectors form an orthogonal set and can be arranged in a matrix  $\mathbf{R}$ , i.e.

$$\mathbf{R} = (\mathbf{u}_0 \ \mathbf{u}_1 \ \mathbf{u}_2)^T \quad (1)$$

where  $A^T$  indicates the transpose of  $A$ .

This matrix will be referred to as the “orientational matrix” in the following. The orientational matrix relates the coordinates  $\mathbf{x}$  in the laboratory reference system to the coordinates  $\mathbf{x}'$  in the HRB reference system via:

$$\mathbf{x}' = \mathbf{R}\mathbf{x} \quad (2)$$

The following discussion focuses on HRBs with finite volume and bounded surface. We assume that the equation of the surface of the HRB, in the “HRB reference system” with origin in the center of mass and axes parallel to the vectors  $\{\mathbf{u}_i\}_i$ , is of the form  $f(\mathbf{r}) = 0$ , where  $f$  changes sign passing from the interior to the exterior of the HRB. Moreover, the normal  $\frac{\partial f}{\partial \mathbf{r}}$  and its first order derivatives are assumed to be properly defined.

### 2.2. Motion of rigid bodies

In the following equations we assume that the three eigenvalues of the inertia tensor are all equal to  $I$ . The formulas for the free rotation of a symmetric-top case are just slightly more elaborated [49], although the free rotation for a general rigid body involves the calculation of special functions [49, 50] and requires a more sophisticated algorithm which has been implemented only recently [51, 52]. Nevertheless, this paper being focused on an algorithm which predicts collision between HRBs, for the sake of simplicity the discussion will be restricted to the case of a fully symmetric inertia tensor. It is straightforward to generalize the approach to a completely asymmetric inertia tensor.

From the angular velocity  $\mathbf{w} = (w_x, w_y, w_z)$  of a free rigid body the antisymmetric matrix  $\mathbf{\Omega}$  can be built, i.e.:

$$\mathbf{\Omega} = \begin{pmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{pmatrix} \quad (3)$$

It is possible to relate the orientation  $\mathbf{R}(t)$  to the orientation at time  $t = 0$  via [49, 53]:

$$\mathbf{R}(t) = \mathbf{R}(0)(\mathbf{I} + \mathbf{M}) \quad (4)$$

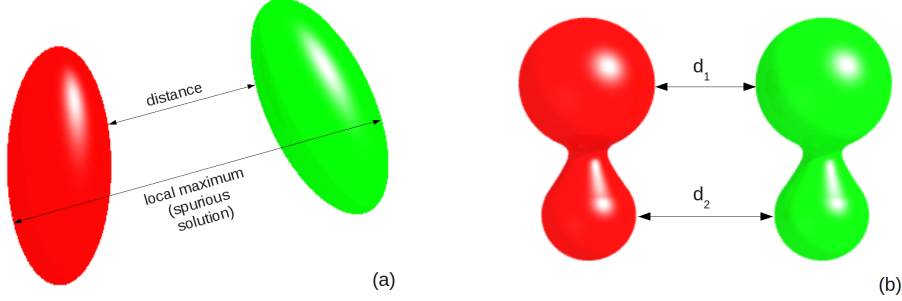


Figure 1: Examples of solution of Eq. 9. (a) Distance between two HRBs with a possible spurious solution. (b) For non-convex objects, there can be multiple solutions.

where  $\mathbf{M}$  is the following matrix:

$$\mathbf{M} = -\frac{\sin(wt)}{w}\mathbf{\Omega} + \frac{1 - \cos(wt)}{w^2}\mathbf{\Omega}^2 \quad (5)$$

with  $w = \|\mathbf{w}\|$ . Note that if  $w = 0$  then  $\mathbf{R}(t) = \mathbf{R}(0)$ .

The update of position and orientation of a free rigid body is therefore accomplished by:

$$\mathbf{x}(t) = \mathbf{x}(0) + \mathbf{v}t \quad (6a)$$

$$\mathbf{R}(t) = \mathbf{R}(0)(\mathbf{I} + \mathbf{M}) \quad (6b)$$

where  $\mathbf{x}(t)$  is the position of the center of mass of the rigid body at time  $t$  and  $\mathbf{v}$  is its velocity.

### 2.3. Distance between two rigid bodies

The present algorithm is based on a calculation of the distance between HRBs. In the following we will show how such a distance is calculated. Consider two rigid bodies,  $A$  and  $B$ , whose surfaces are implicitly defined by the following equations:

$$f(\mathbf{x}) = 0 \quad (7a)$$

$$g(\mathbf{x}) = 0 \quad (7b)$$

It is also assumed that if a point  $\mathbf{x}$  is inside the rigid body  $A$  then  $f(\mathbf{x}) < 0$  ( $g(\mathbf{x}) < 0$ ), while if it is outside  $f(\mathbf{x}) > 0$  ( $g(\mathbf{x}) > 0$ ). The distance  $d$  between these two objects can be defined as follows:

$$d = \min_{\substack{f(\mathbf{x}_A)=0 \\ g(\mathbf{x}_B)=0}} \|\mathbf{x}_A - \mathbf{x}_B\| \quad (8)$$

The latter equation means that the quantity  $\|\mathbf{x}_A - \mathbf{x}_B\|$  has to be minimized with the constraints that points  $\mathbf{x}_A$  and  $\mathbf{x}_B$  belong respectively to the surfaces of  $A$  and  $B$ . Hence,

introducing the Lagrange multipliers  $\alpha$  and  $\beta$ , the distance between the two HRBs  $A$  and  $B$  can be defined as the solution of the following set of 8 equations:

$$f_{\mathbf{x}_A} = -\alpha^2 g_{\mathbf{x}_B} \quad (9a)$$

$$f(\mathbf{x}_A) = 0 \quad (9b)$$

$$g(\mathbf{x}_B) = 0 \quad (9c)$$

$$\mathbf{x}_A + \beta f_{\mathbf{x}_A} = \mathbf{x}_B \quad (9d)$$

where  $\mathbf{x}_A = (x_A, y_A, z_A)$ ,  $\mathbf{x}_B = (x_B, y_B, z_B)$ ,  $f_{\mathbf{x}_A} = \left(\frac{\partial f}{\partial \mathbf{x}_A}\right)^T$  and  $g_{\mathbf{x}_B} = \left(\frac{\partial g}{\partial \mathbf{x}_B}\right)^T$ .

Eqs. (9b) and (9c) guarantee that  $\mathbf{x}_A$  and  $\mathbf{x}_B$  are points on  $A$  and  $B$ , Eq.(9a) guarantees that the normals to the surfaces are anti-parallel, and Eq.(9d) guarantees that the displacement of  $\mathbf{x}_A$  from  $\mathbf{x}_B$  is collinear to the normals to the surfaces.

Equations (9) define extremal points of  $d$  [54]; for example for two convex HRBs local maxima can be found (see Fig. 1(a)) and for two general non-overlapping non-convex HRBs these equations can have multiple solutions (see Fig. 1(b)), although only the smallest one is the actual distance. Therefore, to solve these equations iteratively it is necessary to start from a good initial guess of  $(\mathbf{x}_A, \mathbf{x}_B, \alpha, \beta)$  to avoid finding spurious solutions.

In addition we note that if two rigid bodies overlap slightly (i.e. the overlapped volume is small) there is a solution with  $\beta < 0$ , that is a measure of the inter-penetration of the two rigid bodies; such a solution will be referred to as the “negative distance” solution (Fig.2).

Finally, we define the quantities  $d_i$  as follows:

$$d_i \equiv \|\mathbf{x}_A^i - \mathbf{x}_B^i\| \quad (10)$$

where  $(\mathbf{x}_A^i, \mathbf{x}_B^i, \alpha_i, \beta_i)$  is a solution of Eqs. (9), the distance  $d$  between two HRBs can be formally written as:

$$d = \text{sgn}(\beta_{\min}) \min_i \{d_i\} \quad (11)$$

where  $\text{sgn}(x)$  is the sign function and  $\beta_{\min}$  is the  $\beta_i$  corresponding to the solution with the smallest  $d_i$ .

### 2.3.1. The Newton-Raphson method for calculating the distance

The set of equations (9) can be conveniently solved by a Newton-Raphson (NR) method [55]. This method, as long as a good initial guess is provided, reaches the solution very quickly thanks to its quadratic convergence [55]. If one defines:

$$\mathbf{F}(\mathbf{y}) = \begin{pmatrix} f_{\mathbf{x}_A} + \alpha^2 g_{\mathbf{x}_B} \\ f(\mathbf{x}_A) \\ g(\mathbf{x}_B) \\ \mathbf{x}_A + \beta f_{\mathbf{x}_A} - \mathbf{x}_B \end{pmatrix} \quad (12)$$

---

<sup>1</sup>Note that the gradient  $\frac{\partial}{\partial \mathbf{x}}$  is intended to be a row vector.

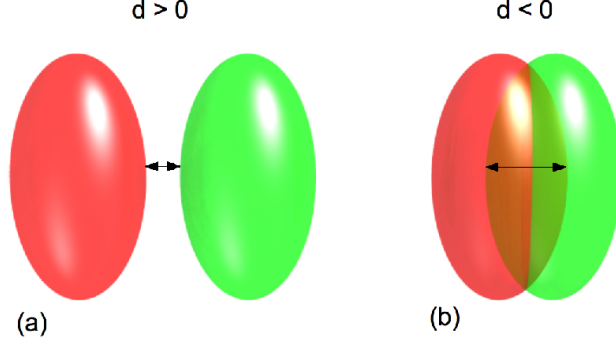


Figure 2: When convex objects slightly overlap, the solution of Eqs.(9) changes sign.

The Eqs. (9) become:

$$\mathbf{F}(\mathbf{y}) = 0 \quad (13)$$

where  $\mathbf{y} = (\mathbf{x}_A, \mathbf{x}_B, \alpha, \beta)$ .

Given an initial point  $\mathbf{y}_0$ , a sequence of points converging to the solution can be built as follows:

$$\mathbf{y}_{i+1} = \mathbf{y}_i + \mathbf{J}^{-1}\mathbf{F}(\mathbf{y}_i) \quad (14)$$

where  $\mathbf{J}$  is the Jacobian of  $\mathbf{F}$ , i.e.:

$$\mathbf{J} = \begin{pmatrix} \frac{\partial f_{\mathbf{x}_A}}{\partial \mathbf{x}_A} & \alpha^2 \frac{\partial g_{\mathbf{x}_B}}{\partial \mathbf{x}_B} & 2\alpha g_{\mathbf{x}_B} & \mathbf{0} \\ f_{\mathbf{x}_A}^T & \mathbf{0}^T & 0 & 0 \\ \mathbf{0}^T & g_{\mathbf{x}_B}^T & 0 & 0 \\ \mathbf{I} + \beta \frac{\partial f_{\mathbf{x}_A}}{\partial \mathbf{x}_A} & -\mathbf{I} & 0 & f_{\mathbf{x}_A} \end{pmatrix} \quad (15)$$

with  $\mathbf{I}$  being the identity 3x3 matrix and  $\mathbf{0}$  the null column vector.

The NR method may not converge if the initial guess is not close enough to the root, hence in general it may be convenient for the sake of numerical robustness to use a globally convergent NR method (see [55]) that converges to the solution from any starting point. An alternative route to ensure the appropriate robustness is to provide an accurate initial guess, e.g. making use of a steepest descent method, as it will be shown in the next section. The matrix inversion, required to evaluate  $\mathbf{J}^{-1}$ , can be performed by standard *LU* decomposition [55]. *LU* decomposition is of order  $N^3/3$ , where  $N$  is the number of equations (8 in the present case). In Sec. 2.3.3 it will be shown that the above set of equations can be reduced to 5, thus reducing the time to invert the matrix by a factor  $\sim 4$ .

### 2.3.2. Initial Guess for the distance: the steepest-descent method.

As initial guess of the NR it may be convenient to choose the closest pair of points on the intersection of the surfaces with the line joining the center of mass of the two

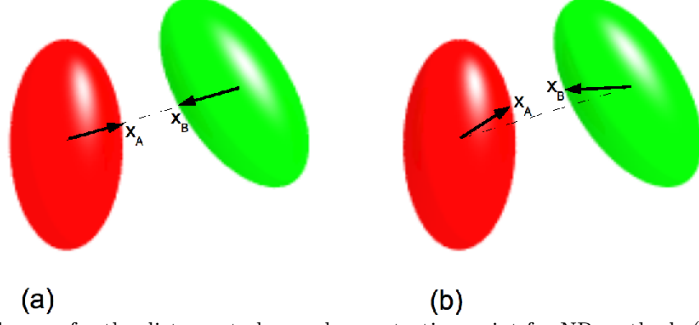


Figure 3: Initial guess for the distance to be used as a starting point for NR method. (a) Simple initial guess for the distance: initial points  $\mathbf{x}_A$  and  $\mathbf{x}_B$  for the NR are obtained considering the interceptions of the line joining the two HEs centers with their surfaces. (b) A better guess for the distance in the case of moderately elongated HRBs (see discussion in Sec. 3.2 for the HEs case).

HRBs (Fig. 3a). Such guess is reasonable if the objects are almost spherical; otherwise a refinement is needed.

A general method to refine such initial guess is to minimize the function

$$D(\mathbf{y}) = \alpha_{SD} \|\mathbf{x}_A - \mathbf{x}_B\|^2 \quad (16)$$

where  $\mathbf{y} = (\mathbf{x}_A, \mathbf{x}_B)$  with the constraints  $f(\mathbf{x}_A) = 0$  and  $g(\mathbf{x}_B) = 0$ , and  $\alpha_{SD}$  is parameter that can be tuned to optimize the steepest-descent (SD). For solving such a problem, steepest descent steps are used, followed by corrections that hold the points  $\mathbf{x}_A$  and  $\mathbf{x}_B$  on the surface of the two bodies. The algorithm is the following one:

1. Choose an initial guess for  $\mathbf{x}_A$  and  $\mathbf{x}_B$
2. Evaluate the gradient  $D_{\mathbf{y}}$  of  $D(\mathbf{y})$ :

$$D_{\mathbf{y}} = (\mathbf{h}_A, \mathbf{h}_B) = 2\alpha_{SD}(\mathbf{x}_A - \mathbf{x}_B, -(\mathbf{x}_A - \mathbf{x}_B)) \quad (17)$$

3. Calculate the components of  $\mathbf{h}_A$  and  $\mathbf{h}_B$  parallel to the surface, i.e.:

$$D_{\mathbf{y}}^{\parallel} = (\mathbf{h}_A^{\parallel}, \mathbf{h}_B^{\parallel}) \quad (18)$$

where

$$\mathbf{h}_{\mu}^{\parallel} = \mathbf{h}_{\mu} - (\mathbf{h}_{\mu} \cdot \hat{\mathbf{n}}_{\mu})\hat{\mathbf{n}}_{\mu} \quad (19)$$

with  $\mu \in \{A, B\}$  and  $\hat{\mathbf{n}}_A = f_{\mathbf{x}_A} / \|f_{\mathbf{x}_A}\|$ ,  $\hat{\mathbf{n}}_B = g_{\mathbf{x}_B} / \|g_{\mathbf{x}_B}\|$

4. Move the two points in the direction of  $D_{\mathbf{y}}^{\parallel}$ :

$$\mathbf{y}' = (\mathbf{x}'_A, \mathbf{x}'_B) = \mathbf{y} - \lambda_{SD} D_{\mathbf{y}}^{\parallel} \quad (20)$$

whith  $0 < \lambda_{SD} < 1$ .



5. Add a small displacement  $d\mathbf{y} = (\xi_A f_{\mathbf{x}_A}, \xi_B g_{\mathbf{x}_B})$  to the vector  $\mathbf{y}'$ , i.e.:

$$\mathbf{y}'' = (\mathbf{x}_A'', \mathbf{x}_B'') = \mathbf{y}' + d\mathbf{y} \quad (21)$$

where  $\xi_A$  and  $\xi_B$  are such that the constraints are satisfied and the two points  $\mathbf{x}_A''$  and  $\mathbf{x}_B''$  belong to the surfaces of the two rigid bodies, i.e.

$$f(\mathbf{x}_A' + \xi_A f_{\mathbf{x}_A}) = 0 \quad (22a)$$

$$g(\mathbf{x}_B' + \xi_B g_{\mathbf{x}_B}) = 0 \quad (22b)$$

6. Terminate if the angle between  $\hat{\mathbf{n}}$  and  $D_{\mathbf{y}}$  is small enough, otherwise go back to step 1.

This procedure provides a guess for  $\mathbf{x}_A$  and  $\mathbf{x}_B$ . Note that the adjustment of the position of the points  $A$  and  $B$  to hold them respectively onto the surfaces  $f$  and  $g$  can be implemented again with a one-dimensional Newton-Raphson method, that will generally converge in few steps if the points are not too far from the surfaces.

The NR method for the distance requires also a guess for  $\alpha$  and  $\beta$  (introduced in Eq. (9));  $\alpha = \|f_{\mathbf{x}_A}\|/\|g_{\mathbf{x}_B}\|$ ,  $\beta = 0$  proved to be a good guess. If the accuracy required for the convergence of the SD is high enough, the NR method will always converge to the correct solution. However, being the SD method much slower than NR, a trade-off between accuracy and speed is needed.

### 2.3.3. Reduced system of equations

The system in Eq. (9) can be reduced to 5 equations eliminating Eq. (9d):

$$f_{\mathbf{x}_A} + \alpha^2 g_{\mathbf{x}_B} (\mathbf{x}_A + \beta f_{\mathbf{x}_A}) = 0 \quad (23a)$$

$$f(\mathbf{x}_A) = 0 \quad (23b)$$

$$g(\mathbf{x}_A + \beta f_{\mathbf{x}_A}) = 0 \quad (23c)$$

In this case the Jacobian is:

$$\mathbf{J} = \begin{pmatrix} \frac{\partial f_{\mathbf{x}_A}}{\partial \mathbf{x}_A} + \alpha^2 \mathbf{A} & 2\alpha g_{\mathbf{x}_B} & \alpha^2 \frac{\partial g_{\mathbf{x}_B}}{\partial \mathbf{x}_B} f_{\mathbf{x}_A} \\ f_{\mathbf{x}_A}^T & 0 & 0 \\ g_{\mathbf{x}_B}^T + \beta g_{\mathbf{x}_B}^T \frac{\partial f_{\mathbf{x}_A}}{\partial \mathbf{x}_A} & 0 & g_{\mathbf{x}_B} \cdot f_{\mathbf{x}_A} \end{pmatrix} \quad (24)$$

where  $\mathbf{A}$  is the following matrix:

$$\mathbf{A} = \frac{\partial g_{\mathbf{x}_B}}{\partial \mathbf{x}_B} + \beta \frac{\partial g_{\mathbf{x}_B}}{\partial \mathbf{x}_B} \frac{\partial f_{\mathbf{x}_A}}{\partial \mathbf{x}_A} \quad (25)$$

Evaluation of  $\mathbf{J}^{-1}$  can be done again by the  $LU$  decomposition.

#### 2.3.4. Relaxed Newton-Raphson method

Eq. (14) has been derived using a first order expansion of  $F(\mathbf{y})$ , but if the matrix  $\mathbf{J}$  is singular or nearly-singular in the proximity of the solution, even with a good initial guess the NR step can become too large and the NR method unstable. To solve this problem, a damping factor into Eq. (14) can be conveniently introduced:

$$\mathbf{y}_{i+1} = \mathbf{y}_i + \xi_i \mathbf{J}^{-1} \mathbf{F}(\mathbf{y}_i) \quad (26)$$

where  $0 < \xi_i < 1$  and a suitable  $\xi_i$  has to be found in order to stabilize the NR method. First, consider either the system of equations (9) or (23); let  $d\mathbf{y}_i = \xi_i (d\mathbf{x}_A, d\mathbf{x}_B, d\alpha^2, d\beta)$  be the step predicted by NR. The relative variations of  $\alpha^2$ ,  $\beta$  can be bounded, imposing the conditions  $\xi_i |d\alpha^2| = \xi_i |\alpha d\alpha| = \epsilon_r \alpha^2$  and  $\xi_i |d\beta| = \epsilon_r |\beta|$  with  $0 < \epsilon_r < 1$ .

Taking the modulus of both sides of Eq.(9a) one gets  $\alpha^2 = \|f_{\mathbf{x}_A}\|/\|g_{\mathbf{x}_B}\|$ , obtaining the condition:

$$\xi_i = \frac{\epsilon_r \|f_{\mathbf{x}_A}\|}{2 \|g_{\mathbf{x}_B}\| |\alpha|} \quad (27)$$

Similarly, from Eq. (9d), one obtains the equation  $\|\mathbf{x}_A - \mathbf{x}_B\| = \|f_{\mathbf{x}_A}\| |\beta|$ . Assuming that  $\|\mathbf{x}_A - \mathbf{x}_B\| \sim l$ , where  $l$  is a typical distance of the system, provides the condition:

$$\xi_i = \frac{\epsilon_r l}{|\beta| \|f_{\mathbf{x}_A}\|} \quad (28)$$

Therefore, the damping factor  $\xi_i$  can be chosen such that:

$$\xi_i = \min \left\{ \frac{\epsilon_r l}{|\beta| \|f_{\mathbf{x}_A}\|}, \frac{\epsilon_r \|f_{\mathbf{x}_A}\|}{2 \|g_{\mathbf{x}_B}\| |\alpha|} \right\} \quad (29)$$

to ensure that the variations of  $\alpha$  and  $\beta$  remain sufficiently small.

#### 2.4. Prediction of the collision time

In an event-driven (ED) algorithm one needs to predict the time of collision of pairs of HRBs. This means to evaluate — given two objects at time  $t = 0$  and the distance as a function of time  $d(t)$  — the smallest time  $t_c > 0$  such that  $d(t_c) = 0$ . If a good guess of the contact point and time is provided, solving a proper set of equations through a NR method (as it will be shown shortly) allows to find the exact contact point and time. In the following it will also be shown how to calculate in a very efficient and simple way such an initial guess, exploiting the evaluation of the distance between two rigid bodies.

##### 2.4.1. Newton-Raphson method for the contact time

The point and time of collision satisfy the following equations:

$$f_{\mathbf{x}}(\mathbf{x}, t) + \alpha^2 g_{\mathbf{x}}(\mathbf{x}, t) = 0 \quad (30a)$$

$$f(\mathbf{x}, t) = 0 \quad (30b)$$

$$g(\mathbf{x}, t) = 0 \quad (30c)$$

where  $f$  and  $g$  now depend also on time because the objects move. Again, it is appropriate to employ the NR method to solve such a system; the Jacobian for the NR in this case turns out to be:

$$\mathbf{J} = \begin{pmatrix} \frac{\partial f_{\mathbf{x}}}{\partial \mathbf{x}} + \alpha^2 \frac{\partial g_{\mathbf{x}}}{\partial \mathbf{x}} & 2\alpha g_{\mathbf{x}} & \frac{\partial f_{\mathbf{x}}}{\partial t} + \alpha^2 \frac{\partial g_{\mathbf{x}}}{\partial t} \\ f_{\mathbf{x}}^T & 0 & f_t \\ g_{\mathbf{x}}^T & 0 & g_t \end{pmatrix} \quad (31)$$

where  $f_t = \frac{\partial f}{\partial t}$  and  $g_t = \frac{\partial g}{\partial t}$ . As discussed in [56], such method becomes very unstable even for simple convex objects unless one starts from a very good initial guess. To construct such a guess, one can find a good bracketing of the collision time, i.e. two times  $t_1, t_2$  such that:

1.  $d(t_1) > 0$  and  $d(t_2) < 0$
2.  $t_2 - t_1$  is small enough to have at the most one collision within  $(t_1, t_2)$
3.  $d(t_1), d(t_2)$  are small enough in order to avoid any instability of the NR method (see above).

Once the latter bracketing has been found, the initial time for the NR is set to  $t_1$ , while a good initial guess for the contact point will be halfway on the distance between the two bodies at  $t_1$ .

#### 2.4.2. Bracketing the contact time

To first bracket the contact time a “bounding sphere” (BS) may be used (Fig.4). The BS of a given HRB  $A$  with center  $\mathbf{r}_A$  is the smallest sphere centered at  $\mathbf{r}_A$  that encloses  $A$ . Given two rigid bodies  $A$  and  $B$  at time  $t = 0$  and their BSs  $C_A$  and  $C_B$ , one must indeed consider three possible cases:

1.  $C_A$  and  $C_B$  do not overlap and they will not collide: in this case  $A$  and  $B$  won't collide either.
2.  $C_A$  and  $C_B$  do not overlap but they will collide: in this case one has to search for a collision within the time interval  $[t_1, t_2]$  where  $t_1$  is the time when the two BSs collide and start overlapping and  $t_2$ , which is greater than  $t_1$ , is the time when they just cease overlapping.
3.  $C_A$  and  $C_B$  overlap: in this case one has to search for a collision within the time interval  $[t_1, t_2]$  where  $t_1 = 0$  and  $t_2 > t_1$  is the time at which the two BSs stop overlapping.

Since the collision prediction between hard spheres (i.e. BSs) is extremely fast this technique improves the performance by reducing the number of collisions to check and provides a first bracketing  $(t_1, t_2)$  of the collision time.

To further improve this bracketing, the following overestimate of the rate of variation of the distance can be established:

$$\dot{d}(t) \leq \|\mathbf{v}_A - \mathbf{v}_B\| + \|\mathbf{w}_A\|L_A + \|\mathbf{w}_B\|L_B \quad (32)$$

where the dot indicates the derivation with respect to time;  $\mathbf{v}_A$  and  $\mathbf{v}_B$  are the velocities of the centers of mass of  $A$  and  $B$ ;  $\mathbf{w}_A$  and  $\mathbf{w}_B$  are the angular velocities of  $A$  and  $B$ , and the lengths  $L_A, L_B$  are such that

$$L_A \geq \max_{f(\mathbf{r}') \leq 0} \{\|\mathbf{r}' - \mathbf{r}_A\|\} \quad (33a)$$

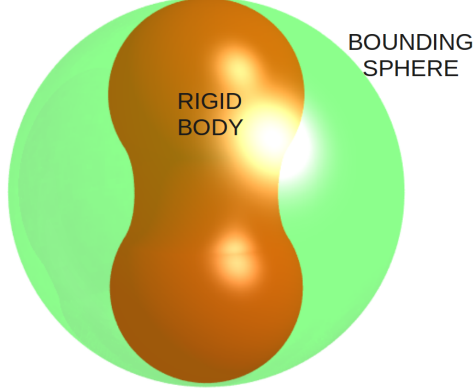


Figure 4: Minimal bounding sphere for a rigid body.

$$L_B \geq \max_{g(\mathbf{r}') \leq 0} \{\|\mathbf{r}' - \mathbf{r}_B\|\} \quad (33b)$$

where  $\mathbf{r}_A, \mathbf{r}_B$  are the centers of mass of the two rigid bodies. To derive Eq.(32), consider the case  $\dot{d}(t) > 0$ . In this case, for a positive time increment  $dt > 0$  one has:

$$|\dot{d}(t)| = \frac{\|\mathbf{x}_A(t+dt) - \mathbf{x}_B(t+dt)\| - \|\mathbf{x}_A(t) - \mathbf{x}_B(t)\|}{dt} \quad (34)$$

where  $\mathbf{x}_A$  and  $\mathbf{x}_B$  are two points belonging to the two rigid bodies such that:

$$d(t) = \|\mathbf{x}_A(t) - \mathbf{x}_B(t)\| \quad (35)$$

If  $\dot{\mathbf{x}}_A(t)$  and  $\dot{\mathbf{x}}_B(t)$  are the velocities at time  $t$  of the points  $\mathbf{x}_A(t)$  and  $\mathbf{x}_B(t)$  respectively, it results that:

$$\|\mathbf{x}_A(t+dt) - \mathbf{x}_B(t+dt)\| \leq \|\mathbf{x}_A(t) - \mathbf{x}_B(t)\| + \|(\dot{\mathbf{x}}_A(t) - \dot{\mathbf{x}}_B(t))dt\| \quad (36)$$

because of Eq. (8), that defines the distance. From Eq. (36) and Eq.(34) the following inequality is obtained:

$$|\dot{d}(t)| \leq \|\mathbf{v}_A - \mathbf{v}_B\| + \|\mathbf{w}_A \times (\mathbf{x}_A(t) - \mathbf{r}_A(t))\| + \|\mathbf{w}_B \times (\mathbf{x}_B(t) - \mathbf{r}_B(t))\| \quad (37)$$

and Eq. (32) results from the fact that:

$$\|\mathbf{x}_A(t) - \mathbf{r}_A(t)\| \leq L_A \quad (38a)$$

$$\|\mathbf{x}_B(t) - \mathbf{r}_B(t)\| \leq L_B \quad (38b)$$

The case  $\dot{d}(t) < 0$  is similar. Indeed, performing the substitutions  $t + dt \rightarrow t'$  and  $dt \rightarrow -dt'$  into Eq. (34), Eq. (32) is obtained again.

With such an overestimate of  $\dot{d}(t)$ , that from now on will be called  $\dot{d}_{max}$ , an efficient strategy for the refinement of the bracketing  $(t_1, t_2)$  of the collision time and point is the following one:

1. Set  $t = t_1$ .
2. Evaluate the distance  $d(t)$  at time  $t$ .
3. Choose a time increment  $\Delta t$  as follows:

$$\Delta t = \begin{cases} \frac{d(t)}{d_{max}}, & \text{if } d(t) > \epsilon_d \\ \frac{\epsilon_d}{d_{max}}, & \text{otherwise.} \end{cases} \quad (39)$$

where  $\epsilon_d \ll \min\{L_A, L_B\}$ .

4. Evaluate the distance at time  $t + \Delta t$ .
5. If  $d(t + \Delta t) < 0$  and  $d(t) > 0$ , then  $t_1 = t$  and  $t_2 = t + \Delta t$ , find the collision time and point via NR and terminate. An initial guess for this NR may be obtained through a quadratic interpolation of the points  $(t, d(t))$ ,  $(t + \Delta t/2, d(t + \Delta t/2))$  and  $(t + \Delta t, d(t + \Delta t))$ .
6. If both  $0 < d(t + \Delta t) < \epsilon_d$  and  $0 < d(t) < \epsilon_d$ , a “grazing” collision may occur between  $t$  and  $t + \Delta t$  (because distance may be first diminishing, then growing). To look for a possible collision, evaluate the distance  $d(t + \Delta t/2)$  and perform a quadratic interpolation of the 3 points  $((t, d(t)), (t + \Delta t/2, d(t + \Delta t/2)), (t + \Delta t, d(t + \Delta t)))$ . There are 3 possible cases:
  - (a) The parabola has a minimum and  $d(t_m) < 0$ , then set  $t_1 = t$  and  $t_2 = t_m$ , find the first zero  $t_z$  of this parabola and use  $t_z$  and  $d(t_z)$  to start a NR to find the collision time and point, and terminate.
  - (b) If the resulting parabola has a minimum  $t_m$  between  $t$  and  $t + \Delta t$ , but  $d(t_m) > 0$ , then find the minimum  $t_{mb}$  of  $d(t)$  between  $t$  and  $t + \Delta t$  with the maximum possible accuracy (using a 1-dimensional Brent’s method for finding minima). If  $d(t_{mb}) < 0$  then set  $t_1 = t$  and  $t_2 = t_{mb}$  and find the first zero  $t_z$  of  $d(t)$  between  $t_1$  and  $t_2$  (with a moderate accuracy, because  $t_z$  will serve only as initial guess for the NR). Use  $t_z$  and  $d(t_z)$  to start a NR to find the collision time and point and terminate.
  - (c) If the parabola does not have a minimum then keep searching (i.e. go to step 7).
7. Increment time by  $\Delta t$ , i.e.  $t \rightarrow t + \Delta t$ .
8. if  $t > t_2$  terminate (no collision has been found).
9. Go to step 2.

Note that if starting at time  $t$  the two rigid bodies collide at time  $t_c > t$  and  $d(t) > \epsilon_d$ , then our over-estimating procedure enforces  $\Delta t < t_c - t$ . If the chosen  $\epsilon_d$  is small enough the distance  $d(t)$  has only one minimum within the time interval  $[t, t + \Delta t]$  and the above scheme for predicting HRBs collisions ensures that “grazing” collisions are properly handled within machine accuracy, i.e. all “grazing” collisions are correctly predicted. According to step 6) in the above scheme, indeed, if  $d(t) > 0$  and  $d(t + \Delta t) > 0$ , a quadratic interpolation is used to search for a negative minimum (i.e. a minimum such that  $d(t_m) < 0$ ). If such a negative minimum can not be found, an attempt to find it is made with the maximum possible accuracy using a suitable numerical method (e.g. *Brent’s method*). We stress that  $\epsilon_d$  must not be chosen with unreasonably tight tolerance not to miss grazing collision within machine accuracy and, although finding the minimum with Brent’s method can be time-consuming, this method is only a second option after the quadratic interpolation, which is faster and finds the minimum in the majority of cases.

### 2.5. Elastic collision of two hard rigid bodies

As two rigid bodies collide, one has to evaluate the new velocities of centers of mass and the new angular velocities after the collision. Imposing the conservation of impulse, angular momentum and energy, the velocities after the collision can be evaluated as follows:

$$\mathbf{v}_A \rightarrow \mathbf{v}_A + m_A^{-1} \Delta p_{AB} \hat{\mathbf{n}} \quad (40a)$$

$$\mathbf{v}_B \rightarrow \mathbf{v}_B - m_B^{-1} \Delta p_{AB} \hat{\mathbf{n}} \quad (40b)$$

$$\mathbf{w}_A \rightarrow \mathbf{w}_A + \Delta p_{AB} I_A^{-1} (\mathbf{r}_A - \mathbf{x}_C) \times \hat{\mathbf{n}} \quad (40c)$$

$$\mathbf{w}_B \rightarrow \mathbf{w}_B - \Delta p_{AB} I_B^{-1} (\mathbf{r}_B - \mathbf{x}_C) \times \hat{\mathbf{n}} \quad (40d)$$

where  $\mathbf{x}_C$  is the contact point,  $\hat{\mathbf{n}}$  is a unit vector perpendicular to both surfaces at  $\mathbf{x}_C$ ,  $I_A$ ,  $I_B$  are the moments of inertia of the two colliding rigid bodies,  $m_A$ ,  $m_B$  are their masses and

$$\Delta p_{AB} = \frac{2(\mathbf{v}_A + \mathbf{w}_A \times (\mathbf{x}_C - \mathbf{r}_A) - \mathbf{v}_B - \mathbf{w}_B \times (\mathbf{x}_C - \mathbf{r}_B)) \cdot \hat{\mathbf{n}}}{m_A^{-1} + m_B^{-1} + I_A^{-1} \|(\mathbf{r}_A - \mathbf{x}_C) \times \hat{\mathbf{n}}\|^2 + I_B^{-1} \|(\mathbf{r}_B - \mathbf{x}_C) \times \hat{\mathbf{n}}\|^2} \quad (41)$$

### 2.6. Linked lists for bounding spheres

Predicting collisions is the most computationally demanding part of an EDMD. In order to speed up a EDMD of hard spheres, one can use linked lists [36] to avoid to check all the  $N^2$  possible collisions among  $N$  objects. In the linked list method, the simulation box is partitioned into  $M^3$  cells and only collisions between particles inside the same cell or its 26 adjacent cells are accounted for. This means also that, whenever an object crosses a cell boundary going from cell  $a$  to a new cell  $b$ , it has to be removed from cell  $a$  and added to cell  $b$ .

As a first step, one can recover the same method using the BSs location as particles. In this case, the cubic box of side  $L$  containing  $N$  identical rigid bodies is divided into  $M^3$  cells so that each cell has side length of the order of the BS diameter  $\sigma_c$ . After that linked lists of these BSs are built and updated as in a ordinary EDMD of hard spheres [36]. To predict collisions of a rigid body  $A$ , one takes into account only the rigid bodies that have their BSs inside the 26 adjacent cells or in the same cell as  $A$  (see [36] for more details).

Unfortunately, BSs are not very efficient if the volume of the BS is much bigger than the volume of the rigid body, as it happens for example for ellipsoids of high elongation [25, 26]. In this case, the number of possible collisions which must be calculated makes such procedure computationally inefficient. In Section 3.3 an alternative and more efficient method for objects with high elongations will be illustrated.

### 2.7. Putting all together: hard rigid bodies event-driven algorithm

In an ED algorithm many events may occur, such as collisions between particles, cell crossing (if one uses linked lists), saving of a system snapshot, output of measured quantities, etc. All these events should be ordered in a calendar so that the next event to happen can be easily retrieved; at the same time, insertion and deletion of events in the calendar should be done as quickly as possible.

One elegant approach was introduced almost thirty years ago by Rapaport [36], who proposed to arrange all the events into an ordered binary tree (calendar of events), so that

insertion, deletion and retrieving of events could be done with an efficiency  $O(\log \mathcal{N})$ ,  $O(1)$  and  $O(\log \mathcal{N})$  respectively, where  $\mathcal{N}$  is the number of events in the calendar. This solution is adopted to handle events calendar in our simulations; all the details of this method can be found in [36].

Considering that all the tools to develop a standard ED algorithm for hard rigid bodies have been illustrated, the algorithm can be resumed as follows:

1. Initialize the events calendar (predict collisions, cell-crossings, etc.).
2. Retrieve next event  $\mathcal{E}$  and set the simulation time to the time of this event.
3. If final time has been reached, terminate.
4. If  $\mathcal{E}$  is a collision between particles  $A$  and  $B$  then:
  - (a) change angular and center-of-mass velocities of  $A$  and  $B$  according to Eqs. (40)
  - (b) remove from calendar all events (collisions, cell-crossings) in which  $A$  and  $B$  are involved.
  - (c) predict and schedule all possible collisions for  $A$  and  $B$ .
  - (d) predict and schedule the two cell-crossings events for  $A$  and  $B$ .
5. If  $\mathcal{E}$  is a cell crossing of a certain rigid body  $A$ :
  - (a) update linked lists accordingly.
  - (b) remove from calendar all events (collisions, cell-crossings) in which  $A$  is involved.
  - (c) predict and schedule all possible collisions for  $A$  using the updated linked lists.
6. go to step 2.

The novel aspect of the present algorithm is in the time-consuming step (4)c, for which we have provided in the previous section the implementation details.

### 3. Hard ellipsoids with an axis of symmetry

In this Section the details about a specific case for which the algorithm has been tested will be provided [42]: hard ellipsoids with an axis of symmetry. Such ellipsoids are characterized by the elongation  $X_0$ , i.e. the ratio of the length of the symmetry axis with respect to any of the other axes. A new efficient implementation of nearest neighbor lists (NNL) for hard ellipsoids will be also discussed and a careful test of the performance of this approach will be shown.

#### 3.1. Evaluation of the distance between two ellipsoids

The surface of a hard ellipsoid can be implicitly defined as follows:

$$(\mathbf{x} - \mathbf{r}_A)^T \mathbf{X}_A (\mathbf{x} - \mathbf{r}_A) = 0 \quad (42)$$

where  $\mathbf{r}_A$  is the position of the center of mass of the ellipsoid and  $\mathbf{X}_A$  is a  $3 \times 3$  positive definite matrix. In particular if at time  $t = 0$  the rigid body reference system coincides to the laboratory reference system it turns out that the free evolving hard ellipsoid surface is:

$$X_A = \mathbf{R}_A^T(t) \mathbf{X}_A(0) \mathbf{R}_A(t) \quad (43)$$

where

$$\mathbf{X}_A = \begin{pmatrix} a^{-1} & 0 & 0 \\ 0 & b^{-1} & 0 \\ 0 & 0 & c^{-1} \end{pmatrix} \quad (44)$$

and  $a, b, c$  are the three semi-axes of the ellipsoids. For hard ellipsoids with an axis of symmetry, the values of two semiaxes are equal.

To evaluate the distance between two ellipsoids  $A$  and  $B$  at a time  $t$  one has to solve either Eq. (9) or Eq. (23) by the Newton-Raphson method. For Eqs. (9) in this particular case the Jacobian becomes:

$$\mathbf{J} = \begin{pmatrix} 2\mathbf{X}_A & 2\alpha^2\mathbf{X}_B & 2\alpha\mathbf{n}_B & \mathbf{0} \\ \mathbf{n}_A^T & \mathbf{0}^T & 0 & 0 \\ \mathbf{0}^T & \mathbf{n}_B^T & 0 & 0 \\ \mathbf{I} + 2\beta\mathbf{X}_A & -\mathbf{I} & 0 & f_{\mathbf{x}} \end{pmatrix} \quad (45)$$

where  $\mathbf{n}_A = 2\mathbf{X}_A(\mathbf{x}_A - \mathbf{r}_A)$ ,  $\mathbf{n}_B = 2\mathbf{X}_B(\mathbf{x}_B - \mathbf{r}_B)$  and

$$\mathbf{X}_\mu = \mathbf{R}_\mu^T(t)\mathbf{X}_\mu(0)\mathbf{R}_\mu(t) \quad (46)$$

with  $\mu \in \{A, B\}$ .

For Eqs. (23) the Jacobian turns out to be:

$$\mathbf{J} = \begin{pmatrix} 2\mathbf{X}^A + \alpha^2\mathbf{A} & 2\alpha\mathbf{n}_B & \mathbf{c} \\ \mathbf{n}_A^T & 0 & 0 \\ \mathbf{n}_B^T + \mathbf{d}^T & 0 & \mathbf{n}_B \cdot \mathbf{n}_A \end{pmatrix} \quad (47)$$

where  $\mathbf{n}_A$ ,  $\mathbf{n}_B$  and  $\mathbf{X}_\alpha$  are the same as before,  $\mathbf{x}_B = \mathbf{x}_A + \beta\mathbf{n}_A$  and

$$\mathbf{c} = 2\alpha^2 \mathbf{X}_B \cdot \mathbf{n}_A \quad (48a)$$

$$\mathbf{d} = 2\beta \mathbf{n}_B^T \cdot \mathbf{X}_A \quad (48b)$$

$$\mathbf{A} = 2\alpha^2\mathbf{X}_B \cdot (2\beta\mathbf{X}_A + \mathbf{I}) \quad (48c)$$

### 3.2. A better guess of the distance

As already discussed, the NR method needs a good guess of the starting point and for this purpose a steepest descent method has been presented in Sec. 2.3.2. For ellipsoids with moderate elongations ( $0.2 < X_0 < 5.0$ ), the initial guess can be calculated in an alternative and very simple way. The simplest possibility is to use as a first guess for  $\mathbf{x}_A$  and  $\mathbf{x}_B$  the intersections of the vector  $\mathbf{r}_{AB} = \mathbf{r}_A - \mathbf{r}_B$ , joining the centers of mass of the two ellipsoids, with their surfaces. This guess is quite rough and a possible improvement can be achieved, as explained in the following.

First of all the components of  $\mathbf{r}_{AB}$  in the reference systems of the two ellipsoids are calculated, i.e.

$$\mathbf{v}'_A = -\mathbf{R}_A \cdot \mathbf{r}_{AB} \quad (49a)$$

$$\mathbf{v}'_B = \mathbf{R}_B \cdot \mathbf{r}_{AB} \quad (49b)$$

then these two vectors will be scaled by the semi-axes of the ellipsoids and their components will be transformed back to the laboratory reference system, i.e.:

$$\mathbf{v}_A = \mathbf{R}_A^T(\mathbf{S} \cdot \mathbf{v}'_A) \quad (50a)$$

$$\mathbf{v}_B = \mathbf{R}_B^T(\mathbf{S} \cdot \mathbf{v}'_B) \quad (50b)$$



where  $\mathbf{R}_A$  and  $\mathbf{R}_B$  are the orientational matrices of  $A$  and  $B$  respectively and

$$\mathbf{S} = \frac{1}{\sqrt{a^2 + b^2 + c^2}} \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \quad (51)$$

Finally the intersections with the HE surfaces of the vector  $\mathbf{v}_A$  centered at the center of mass of  $A$  and  $\mathbf{v}_B$  centered at the center of mass of  $B$  are computed and these two points are the desired guesses for  $\mathbf{x}_A$  and  $\mathbf{x}_B$  for the NR. The effects of such transformations are shown in the right panel of Fig.(3).

At this point, distance evaluation can be started with the initial values for  $\alpha$  and  $\beta$  set to 0. This method provides a speed-up of around 10%.

### 3.3. Nearest Neighbour Lists

Increasing the elongation of the ellipsoids, the linked list method illustrated before becomes progressively less efficient at moderate and high densities [25]. Indeed given one ellipsoid  $A$ , when using the linked lists one has to predict the collision times of  $A$  with all ellipsoids in the cell of  $A$  and with all ellipsoid in the 26 adjacent cells. In the case of rotationally symmetric ellipsoids, the number of collision time predictions grows as  $X_0^2$  if  $X_0 > 1$  and as  $1/X_0$  if  $X_0 < 1$  [26] (see also Sec. 4.3). In order to reduce the number of predictions at very high or very small elongations, Donev *et al.* [25] suggest to surround each particle  $A$  with a *bounding neighborhood* having the same shape as  $A$  (i.e. in the case of HEs they use ellipsoids) and to predict collisions only between particles having overlapping bounding neighborhoods. Similarly to what is proposed by Donev *et al.* [25] we suggest to build an oriented bounding parallelepiped (OBP), instead of an ellipsoid, at a certain time  $t$  around each HE and to predict collisions only between HEs having overlapping parallelepipeds (Fig. 5).

Each parallelepiped encloses the corresponding ellipsoid completely, and it is centered at the center of mass of the ellipsoid itself. More precisely, given an ellipsoid  $A$  with semi-axes  $a, b, c$  and center of mass  $\mathbf{r}_A$ , the vertices  $\mathbf{v}_\alpha$  of the parallelepiped, with  $\alpha \in \{1 \dots 6\}$  are:

$$\mathbf{v}_\alpha = \mathbf{r}_A + \sigma_2(\alpha)(s_{\sigma_1(\alpha)} + \Delta_{NNL})\mathbf{u}_{\sigma_1(\alpha)} \quad (52)$$

where

$$\begin{aligned} \sigma_1(\alpha) &= 1 + (\alpha - 1) \div 2 \\ \sigma_2(\alpha) &= 2(\alpha \bmod 2) - 1 \end{aligned} \quad (53)$$

and  $\mathbf{s} = (s_1, s_2, s_3) = (a, b, c)$ . Moreover  $\{\mathbf{u}_i\}_{i \in \{1,2,3\}}$  are the principal axes of the ellipsoid, i.e. they are such that:

$$\mathbf{X}_A \mathbf{u}_\alpha = s_\alpha \mathbf{u}_\alpha \quad (54)$$

and  $\Delta_{NNL}$  is a positive parameter that can be tuned to optimize the performance of the NNL (see Sec. 4.5).

Given an ellipsoid  $A$ , the set of ellipsoids having their parallelepipeds overlapping with the parallelepiped enclosing  $A$ , is the NNL of  $A$ . Each parallelepiped  $i$  is immobile and contains only the  $i$ -th ellipsoid; if  $t_i$  is the time when the ellipsoid will collide with its containing parallelepiped, its NNL will have to be rebuilt not after the time  $t_r = \min_i \{t_i\}$ .

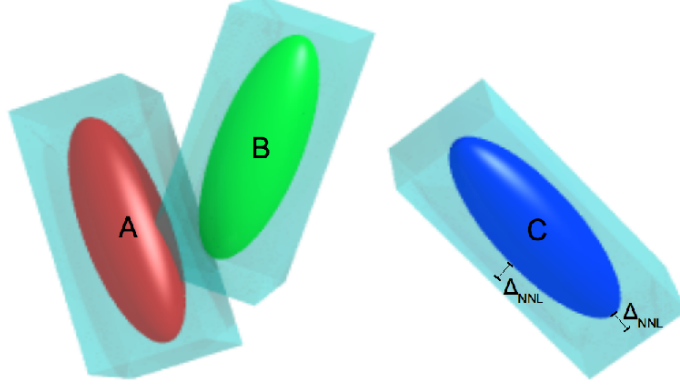


Figure 5: Nearest neighbour lists are built through OBPs: ellipsoids A and B are nearest neighbors, and they may collide among them before colliding with their OBP; neither A or B can collide with C during this time. The "thickness" of the OBP is  $\Delta_{NNL}$ .

In addition, if a collision between two ellipsoids  $i$  and  $j$  occurs, then the new contact times  $t_i$  and  $t_j$  of these two ellipsoids with their parallelepipeds will have to be evaluated and the new time for rebuilding the NNL lists, i.e.  $t'_r = \min\{t_r, t_i, t_j\}$ , will have to be set.

### 3.3.1. Distance between a rigid body and a plane

For predicting the time of collision between an ellipsoid and a parallelepiped the distance between an ellipsoid and each of the 6 faces of the parallelepiped must be calculated. This means that one has to evaluate the distance between the surfaces defined through Eqs. (42) and a plane defined through the following equation:

$$\mathbf{n}_P \cdot (\mathbf{x} - \mathbf{r}_P) = 0 \quad (55)$$

where both  $\mathbf{r}_P$  and  $\mathbf{n}_P$  do not depend on time because the plane is immobile (Fig. 6).

Again a NR method can be used to calculate this distance. In Sec. 2 the second rigid body  $B$  defined by Eq. (7b) may also be a plane (as defined through Eq. (55)) or, alternatively, a plane can be thought as a limiting case of a very large ellipsoids, in which case the Jacobian needed to solve Eqs. (9) turns out to be:

$$\mathbf{J} = \begin{pmatrix} 2\mathbf{X}_A & \mathbf{0} & -2\alpha\mathbf{n}_P & \mathbf{0} \\ \mathbf{n}_A^T & \mathbf{0}^T & 0 & 0 \\ \mathbf{0}^T & \mathbf{n}_P^T & 0 & 0 \\ \mathbf{I} & -\mathbf{I} & 0 & \mathbf{n}_P \end{pmatrix} \quad (56)$$

while the Jacobian for solving Eqs. (23) is:

$$\mathbf{J} = \begin{pmatrix} 2\mathbf{X}_A & -2\alpha\mathbf{n}_P & \mathbf{0} \\ \mathbf{n}_A^T & 0 & 0 \\ \mathbf{n}_P^T & 0 & \mathbf{n}_P \cdot \mathbf{n}_P \end{pmatrix} \quad (57)$$

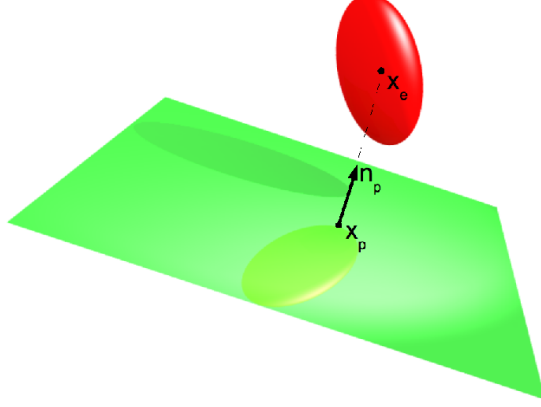


Figure 6: The initial NR guess to calculate the distance between a plane  $P$  and an ellipsoid  $E$  uses the points  $\mathbf{x}_E$  and  $\mathbf{x}_P$ . Here  $\mathbf{n}_P$  is the normal to the plane; the center-of-mass of the ellipsoid is on the line from  $\mathbf{x}_P$  with direction  $\mathbf{n}_P$ .

A good guess is required also to solve Eqs. (9) or Eqs. (23) through a NR method. Let  $l$  be the axis passing through the center of the ellipsoid and parallel to the vector  $\mathbf{n}_P$ . If intersections points  $\mathbf{x}_E$  and  $\mathbf{x}_P$  of  $l$  with the ellipsoid and the plane respectively are evaluated, then these two points (if Eqs. (9) are used) or only  $\mathbf{x}_E$  (if Eqs. (23) are used) proved to be a good initial guess for the NR method.

### 3.3.2. Prediction of the collision time

Given a good bracketing of collision time of an ellipsoid  $A$  with one of the six planes, Eqs. (30), where  $f$  defines the surface of an ellipsoid and  $g$  the surface of a plane, can be solved numerically. The Jacobian to use in the NR is the following:

$$\mathbf{J} = \begin{pmatrix} 2\mathbf{X}_A & -2\alpha\mathbf{n}_P & \mathbf{h} \\ \mathbf{n}_A^T & 0 & k \\ \mathbf{n}_P^T & 0 & 0 \end{pmatrix} \quad (58)$$

where  $\mathbf{h} = \tilde{\Omega} \tilde{\mathbf{x}} - \mathbf{X}_A \mathbf{v}_A$ ,  $k = -\mathbf{v}_A^T \mathbf{X}_A \tilde{\mathbf{x}} + \tilde{\mathbf{x}}^T \tilde{\Omega} \tilde{\mathbf{x}} - \tilde{\mathbf{x}} \mathbf{X}_A \mathbf{v}_A$  and  $\tilde{\Omega} = \mathbf{\Omega}_A \mathbf{X}_A - \mathbf{X}_A \mathbf{\Omega}_A$ ,  $\tilde{\mathbf{x}} = \mathbf{x} - \mathbf{r}_A$ . Here  $\mathbf{\Omega}_A$  is the matrix  $\mathbf{\Omega}$  defined in Eq. (3) corresponding to the angular velocity  $\mathbf{w}_A$  of the ellipsoid  $A$  and  $\mathbf{v}_A$  is the center of mass velocity of  $A$ .

As for the collision of two ellipsoids, the collision time can be bracketed using an overestimate of  $\dot{d}(t)$  (being  $d(t)$  the distance between an ellipsoid and a plane of the parallelepiped). If the ellipsoid surface is the locus of points such that  $f(\mathbf{x}) = 0$  and the plane is defined as in Eq. (55), the distance between an ellipsoid and a plane can be defined as follows:

$$d = \min_{f(\mathbf{x}_A)=0} |\mathbf{x}_A \cdot \hat{\mathbf{n}}_P| \quad (59)$$

where  $\hat{\mathbf{n}}_P = \mathbf{n}_P / \|\mathbf{n}_P\|$ . Similarly to what has been done for two ellipsoids, it can be proved that

$$\dot{d}(t) \leq |\mathbf{v}_A \cdot \hat{\mathbf{n}}_P| + L_A \|\mathbf{w}_A\| \quad (60)$$

From now on, the latter overestimate will be called  $\dot{d}_{pm}$ . It is now possible to illustrate the algorithm to find the collision time of an ellipsoid with its OBP. Consider an ellipsoid  $A$  and the six planes of the corresponding parallelepiped labelled by  $\{p_i\}_{1 \leq i \leq 6}$ . With respect to each plane there are six different overestimations of  $\dot{d}(t)$ , labeled as  $\dot{d}_{pm}^i$ , and six different distances  $d_i(t)$  at time  $t$ . The algorithm is the following one:

1. set  $t = t_1$ .
2. Evaluate the distances  $d_i(t)$  at time  $t$ .
3. Choose a time increment  $\Delta t$  as follows:

$$\Delta t = \begin{cases} \min_i \{d_i(t)/\dot{d}_{pm}^i\}, & \text{if } d_i(t) > \epsilon_d^{nnl}; \\ \frac{\epsilon_d^{nnl}}{\max_i \{\dot{d}_{pm}^i\}}, & \text{otherwise.} \end{cases} \quad (61)$$

where  $\epsilon_d^{nnl} \ll L_A$ .

4. Evaluate the distance at time  $t + \Delta t$ .
5. If for at least one  $i$  one has  $d_i(t + \Delta t) < 0$  and  $d_i(t) > 0$ , then the solution is bracketed. Find the collision time and the collision point through the NR. For the initial guess of the collision time, use the first zero of the parabola resulting from a quadratic interpolation of points  $(t, d(t))$ ,  $(t + \Delta t/2, d(t + \Delta t/2))$  and  $(t + \Delta t, d(t + \Delta t))$ . After that, terminate.
6. For all distances such that  $0 < d_i(t + \Delta t) < \epsilon_d$  and  $0 < d_i(t) < \epsilon_d$ , evaluate the distance  $d_i(t + \Delta t/2)$ , perform a quadratic interpolation of these 3 points  $(t, d_i(t))$ ,  $(t + \Delta t/2, d_i(t + \Delta t/2))$ ,  $(t + \Delta t, d_i(t + \Delta t))$  and find if the resulting parabola has any zeros. If this is the case, refine the position of the smallest zero using again the NR.
7. Increment time by  $\Delta t$ , i.e.  $t \rightarrow t + \Delta t$ .
8. if time is greater than  $t_2$ , terminate (no collisions have been found).
9. Go to step 2.

Grazing collisions in this specific case are dealt with differently from collisions between two HEs. The trick is to consider, for predicting the escape time of a HE, a smaller bounding box. In particular the chosen distance between each of the 6 planes forming the bounding box and the embedded HE is  $\Delta_{NNL} - \epsilon_d^{nnl}$ . With this choice of the bounding box dimensions, if a collision is missed, due to a grazing collision, the HE is not outside its bounding box anyway and values for  $\epsilon_d^{NNL}$  around  $10^{-2}b$  can be safely chosen.

The times  $t_1$  and  $t_2$  are evaluated making use of the BS of  $A$  as illustrated in the following. Consider the ellipsoid  $A$  and its BS  $C_A$  at a certain time  $t$  when the NNL has to be rebuilt. Two possible different cases may occur:

1.  $C_A$  is enclosed into the parallelepiped: in this case  $t_1$  is the time when the BS will collide with one of the six planes of the parallelepiped and  $t_2 > t_1$  is the time when the BS stops intersecting the plane.
2.  $C_A$  intersects the parallelepiped: in this case  $t_1 = t$  and  $t_2 > t_1$  is the time at which the two BSs cease to overlap.

### 3.3.3. Overlap of Parallelepipeds

For building NNL all the parallelepipeds overlapping with a given one have to be found. This task can be accomplished quite efficiently, using a technique well known in computer graphics [57]. Consider two parallelepipeds corresponding to ellipsoids  $A$  and  $B$  with centers  $\mathbf{r}_A$  and  $\mathbf{r}_B$ , principal axes  $\mathbf{u}_\alpha^A$  and  $\mathbf{u}_\alpha^B$  and semi-axes  $a, b, c$ .

Consider the following straight lines originating from the center of  $A$ :

$$\mathbf{t}_j = \mathbf{r}_A + \xi \mathbf{w}_j \quad (62)$$

with

$$\mathbf{w}_j \in \{\{\mathbf{u}_\alpha^A\}_\alpha, \{\mathbf{u}_\alpha^B\}_\alpha, \{\mathbf{u}_\alpha^A \times \mathbf{u}_\beta^B\}_{\alpha,\beta}\} \quad (63)$$

where  $\alpha, \beta = 1, 2, 3$ ,  $\xi \in \mathbb{R}$  and  $j = 1 \dots 15$  labels all the possible directions.

The centers of the ellipsoids  $\mathbf{r}_A$  and  $\mathbf{r}_B$  will be projected onto all these lines, obtaining the points  $\tilde{\mathbf{r}}_j^A$  and  $\tilde{\mathbf{r}}_j^B$  and the vertices of the two parallelepipeds will be projected as well, obtaining the points  $\mathbf{p}_{i,j}^A$  and  $\mathbf{p}_{i,j}^B$ , where  $i = 1 \dots 8$  labels all the possible vertices of a parallelepiped.

Using the projected vertices one can build for each direction  $j$  two spheres with centers  $\tilde{\mathbf{r}}_j^A$  and  $\tilde{\mathbf{r}}_j^B$  and radii:

$$\begin{aligned} \rho_j^A &= \max_i \{\|\mathbf{p}_{i,j}^A - \tilde{\mathbf{r}}_j^A\|\} \\ \rho_j^B &= \max_i \{\|\mathbf{p}_{i,j}^B - \tilde{\mathbf{r}}_j^B\|\} \end{aligned} \quad (64)$$

The two parallelepipeds do not overlap if and only if all these pairs of spheres do not overlap.

### 3.4. Prediction of the collision time between two ellipsoids

Exploiting linked lists and BSs, one obtains a time interval  $[t_1, t_2]$  that brackets the possible contact time between two ellipsoids  $A$  and  $B$ . Making also use of NNL, a better bracketing of the collision time can be estimated, with the bracketing time interval becoming  $[t_1, \tilde{t}_2]$ , with:

$$\tilde{t}_2 = \min\{t_2, t_r\} \quad (65)$$

where  $t_r$  is the time at which the NNL rebuild is scheduled. Then starting from  $t = t_1$  the collision time can be finely bracketed by applying the algorithm illustrated in Sec. 2.7 for the general case of hard rigid bodies with the substitution  $t_2 \rightarrow \tilde{t}_2$ .

The bracketing of the contact point and time for the collision between two HEs can be used in the NR to solve Eqs. (30) with the following Jacobian:

$$\mathbf{J} = \begin{pmatrix} 2(\mathbf{X}_A + \alpha^2 \mathbf{X}_B) & -2\alpha \mathbf{n}_B & \mathbf{h} \\ \mathbf{n}_A^T & 0 & k_A \\ \mathbf{n}_B^T & 0 & k_B \end{pmatrix} \quad (66)$$

where

$$\mathbf{h} = \tilde{\Omega}_A \tilde{\mathbf{x}}_A - \mathbf{x}_A \mathbf{v}_A + \alpha^2 (\tilde{\Omega}_B \tilde{\mathbf{x}}_B - \mathbf{x}_B \mathbf{v}_B) \quad (67)$$

and

$$k_A = -\mathbf{v}_A^T \mathbf{X}_A \tilde{\mathbf{x}}_A + \tilde{\mathbf{x}}_A^T \tilde{\Omega}_A \tilde{\mathbf{x}}_A - \tilde{\mathbf{x}}_A^T \mathbf{X}_A \mathbf{v}_A$$

$$k_B = -\mathbf{v}_B^T \mathbf{X}_B \tilde{\mathbf{x}}_A + \tilde{\mathbf{x}}_A^T \tilde{\Omega}_B \tilde{\mathbf{x}}_B - \tilde{\mathbf{x}}_B^T \mathbf{X}_B \mathbf{v}_B \quad (68)$$

Here, one has:  $\tilde{\Omega}_A = \Omega_A \mathbf{X}_A - \mathbf{X}_A \Omega_A$ ,  $\tilde{\Omega}_B = \Omega_B \mathbf{X}_B - \mathbf{X}_B \Omega_B$ ,  $\tilde{\mathbf{x}}_A = \mathbf{x} - \mathbf{r}_A$ ,  $\tilde{\mathbf{x}}_B = \mathbf{x} - \mathbf{r}_B$ , where  $\Omega_A$  is the matrix  $\Omega$  defined in Eq. (3) corresponding to the angular velocity  $\mathbf{w}_A$  of the ellipsoid  $A$ , and  $\Omega_B$  is the same matrix for the ellipsoid  $B$ .

### 3.5. Event-driven molecular dynamics of Hard Ellipsoids

The ED molecular dynamics of HEs is similar to what illustrated in Sec. 2.7. The only addition is the use of NNL to speed the simulation up. In the following the scheme of an EDMD of HEs is given:

1. Initialize the events calendar (predict collisions, cells crossings, etc.).
2. Retrieve next event  $\mathcal{E}$  and set the simulation time to the time of this event.
3. If final time has been reached, terminate.
4. If  $\mathcal{E}$  is the “NNL rebuild” event then:
  - (a) remove all events from calendar.
  - (b) update the system to current time.
  - (c) evaluate  $t_r$ .
  - (d) check for overlaps between parallelepipeds and build the NNL accordingly.
  - (e) predict all cell-crossings and schedule them.
  - (f) predict all collisions between ellipsoids and schedule them.
  - (g) schedule next NNL rebuild.
  - (h) schedule all other remaining events removed from calendar (output of summary, etc.).
5. If  $\mathcal{E}$  is a collision between particles  $A$  and  $B$  then:
  - (a) change angular and center-of-mass velocities of  $A$  and  $B$  according to Eqs. (40)
  - (b) remove from calendar all events (collisions, cell-crossings) in which  $A$  and  $B$  are involved.
  - (c) evaluate new collision times  $t_A$  and  $t_B$  of  $A$  and  $B$  with their parallelepipeds. and schedule a new “NNL rebuild” event if  $t_A$  or  $t_B$  are less than  $t_r$ .
  - (d) predict and schedule the two cell crossings events for  $A$  and  $B$ .
  - (e) predict and schedule all possible collisions for  $A$  and  $B$  using the NNL of  $A$  and  $B$ .
6. If  $\mathcal{E}$  is a cell crossing of a certain rigid body  $A$ :
  - (a) update linked lists accordingly.
  - (b) remove from calendar all events (collisions, cell-crossings) in which  $A$  is involved.
  - (c) predict and schedule all possible collisions for  $A$  using the NNL of  $A$ .
7. Go back to step 2.

Note that linked lists are used just to rebuild the NNL, that is given a parallelepiped corresponding to an ellipsoid  $A$ , one searches for overlapping parallelepipeds among all the ones that are in the same cell as  $A$  or in one of the 26 neighbors cells, assuming that the cells side is greater than the length of the diagonal of the parallelepipeds. To rebuild NNLs Donev *et al.* [26] use, in addition to LL, also a collection of small spheres, called *bounding sphere complex* and in their implementation this method ensures that the cost of building the NNLs can be controlled increasing the elongation. In our implementation of NNL, that relies on the use of bounding parallelepipeds, the average collision time

(see discussion in Section 4.3) is quite independent from elongation for  $X_0$  up to 10. This is mainly due to the fact that the time needed to check overlaps between bounding parallelepipeds is negligible.

In the above scheme NNL update is a global event (complete update), in [26] a method is described that allows to update only NNLs of affected particles after a collision (partial update). In their implementation NNL are built using HEs instead of parallelepipeds and partial update method outperforms the complete update method (see [26]). Differently in our implementation the time needed to update NNLs at moderate and high densities, which is what we are interested in the most, is negligible (see next Section) and the NNLs partial update method cannot offer any significant speedup.

#### 4. Performance results.

To analyze the performance of the algorithm, monodisperse uniaxial HEs in the isotropic liquid phase have been simulated, characterized by the aspect ratio  $X_0 = a/b$  (where  $a$  is the length of the revolution axis,  $b$  of the other ones) and by the packing fraction  $\phi = \pi X_0 b^3 \rho / 6$  (where  $\rho = N/V$  is the number density).  $N = 256$  particles have been simulated at various volumes  $V$  and elongations  $X_0$  in a cubic box with periodic boundary conditions. The length of the smallest semi-axis is chosen as unit of distance and the mass  $m$  of the particle as unit of mass ( $m = 1$ ). Temperature is one in units of the Boltzmann constant  $k_B$ ; the corresponding unit of time is  $\sqrt{ml^2/(k_B T)}$ . A spherically symmetric moment of inertia, i.e.  $I_x = I_y = I_z = 2mr^2/5$  is chosen, where  $r = \min(a, b)/2$  is the radius of the sphere inscribed in the ellipsoid. Notice that the value of the moment of inertia along the symmetry axis is not relevant for the present system, since angular velocity around the symmetry axis is zero. Although ellipsoids of revolution behave as symmetric tops, the colliding surfaces will be treated as perfectly smooth (see Eq. (40)) and therefore the component of angular velocity along the symmetry axis will be conserved in each collision. In the following we indicate as “reduced time” the simulation time expressed in reduced units, while “CPU time” means the real time spent by the CPU for calculations, expressed in seconds. To test the algorithm speed, we use the CPU time per ellipsoid collision, labelled with  $\tau_c$  and defined as:

$$\tau_c = \frac{T_{tot}}{N_{coll}} \quad (69)$$

where  $T_{tot}$  is the (real) time needed to perform  $N_{coll}$  collisions during a simulation.

Predicting a collision for a pair of colliding HEs (labelled by  $A$  and  $B$ ) requires the CPU time  $\Delta t_{coll}$ :

$$\Delta t_{coll} = \sum_l^{N_{pc}^A} \delta t_l^A + \sum_l^{N_{pc}^B} \delta t_l^B \quad (70)$$

where  $N_{pc}^X$  indicates the number of collisions that have to be predicted for ellipsoid  $X$  ( $X = A, B$ ) and with  $\delta t_l^X$  the CPU time requested to predict one collision. Indeed, after a collision between two HEs, all the possible future events involving the two colliding HEs must be predicted. If the quantity “average prediction time” is defined as follows:

$$\langle \delta t \rangle_X \equiv \frac{1}{N_{pc}^X} \sum_l^{N_{pc}^X} \delta t_l^X \quad (71)$$

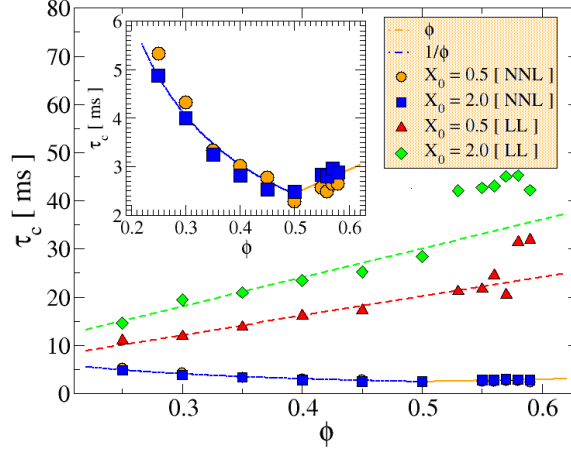


Figure 7: Time per collision  $\tau_c$  in the prolate case ( $X_0 = 0.50$ ) using only LLs (green diamonds) and also NNLs (orange circles) and in the oblate case ( $X_0 = 2.00$ ) using only LLs (red triangles) and also NNLs (blue squares). Dot-dashed lines are guides to the eyes, showing the  $\phi$  and  $1/\phi$  behavior for the NNLs case, while dashed lines are guide to the eyes showing a linear behaviour in  $\phi$  for the LLs case (see discussion in Section 4.2). NNLs show much less density dependence than LLs. Note that in all simulations the number of LL cells into which the box is partitioned is not kept fixed. As a consequence  $\tau_c$  vs  $\phi$  shows a simple linear dependence up to  $\phi \approx 0.55$ , where, due to increasing density, the number of LL cells decreases by one unit breaking the linear behavior of  $\tau_c(\phi)$ . The inset is a zoom of  $\tau_c$  vs  $\phi$  employing also NNLs.

where again  $X = A, B$ . Eq. (70) can be rewritten as:

$$\Delta t_{coll} = N_{pc}^A \langle \delta t \rangle_A + N_{pc}^B \langle \delta t \rangle_B \quad (72)$$

On passing note that, using LL and/or NNL,  $N_{pc}^A$  and  $N_{pc}^B$  are  $O(1)$  varying the total number  $N$  of HEs in the simulation. Such a number will be generally greater for LL than for NNL, but we will discuss this point more accurately later. Also note that  $\langle \delta t \rangle_A$  and  $\langle \delta t \rangle_B$  depend mostly on the average number of Newton-Raphson steps performed to predict the collision time. In addition, during the simulation NNL and LL have to be updated and this will cost an extra time  $\Delta t_u$  per HE collision, so that the time per ellipsoid collision  $\tau_c$  is:

$$\tau_c = \Delta t_{coll} + \Delta t_u \quad (73)$$

The contribution of the LLs update to  $\Delta t_u$  compared to the NNLs update contribution is always negligible, meaning that the main contribution comes from the update of the NNLs, because it is more time consuming. Indeed, NNLs update requires the evaluation of the escape time of each HE from its bounding box and this is time consuming. Moreover, at high densities,  $\Delta t_u$  is usually negligible with respect to  $\Delta t_{coll}$ , because the diffusivity of the particles is low at high densities and the average escape time of HE from their bounding boxes becomes quite long. All simulations are performed on a Intel Xeon CPU 3.06 GHz (codenamed “Prestonia”) with a L2 cache size of 512 KB and a 533 Mhz FSB.



#### 4.1. Generation of the initial configurations

To create the starting configuration at a desired  $\phi$ , an extremely diluted  $\alpha$ -FCC crystal has been melted [58]; afterwards, the particles have been grown independently up to the desired packing fraction (quench in  $\phi$  at fixed  $N$ ,  $X_0$ ). The details of the growth algorithm will be illustrated in a future publication. To generate history-independent configurations, tests have been performed on equilibrium configurations. To test the equilibration of the sample, the decay of the self correlation function  $F_{self}(q, t) = \frac{1}{N} \sum_i e^{i\mathbf{q} \cdot (\mathbf{r}_i(t) - \mathbf{r}_i(0))}$  and of the orientational correlation function,  $C_2(t) = P_2(\cos \theta(t))$  - where  $P_2(x) = (3x^2 - 1)/2$  and  $\theta(t)$  is the angle between the symmetry axis at time  $t$  and at time zero, have been studied.

#### 4.2. Dependence of the algorithm speed on density.

Elongations  $X_0 = 0.50$  and  $2.00$  have been considered, fixing  $\epsilon_d = 10^{-5}$  and the thickness of the neighbour shell  $\Delta_{NNL} = 0.15$ .

Using only LLs both in the prolate and in the oblate case (see Fig. 7)),  $\tau_c$  increases going from lower to higher densities. In this case  $\Delta t_u$  is negligible and the average time to predict a collision  $\langle \delta t \rangle$  is roughly proportional to the number of HEs inside each LL cell, that in turn is proportional to the volume fraction  $\phi$  of the system.

When using both LLs and NNLs the scenario is quite different. In Fig. 7 results from simulations using NNLs are shown, and it is apparent that there are two different regimes of  $\tau_c(\phi)$  at low and high volume fractions  $\phi$ . The interpretation of this observed behavior is quite straightforward. Indeed, considering Eq. (73), below  $\phi \approx 0.5$  there are few HEs within each NNL, i.e.  $N_{pc}^A \approx 0$  and  $N_{pc}^B \approx 0$  and the time per collision  $\tau_c$  is dominated by  $\Delta t_u$ , that is by the time per collision needed to update the NNLs. In this case  $\Delta t_u$  is proportional to the number of average evaluations of the contact time of a HE with its bounding box  $t_r$ , i.e. if  $l \propto 1/\phi$  is roughly the mean free path between two HEs collisions:

$$\Delta t_u \propto l / \Delta_{NNL} \propto \frac{1}{\phi} \quad (74)$$

In Fig. 7 the  $\phi^{-1}$  dependence is evident at small  $\phi$ .

In contrast, at high densities  $\Delta t_u$  is negligible and the CPU time needed to predict a collision is dominated by  $\Delta t_{coll}$ , that in turn depends linearly on the number of neighbors within the bounding box of a certain HE, i.e.

$$\tau_c \propto N_A \propto \frac{4\pi}{3} \phi (a + \Delta_{NNL})(b + \Delta_{NNL})(c + \Delta_{NNL}) - 1 \quad (75)$$

At high volume fractions this equation simplifies to  $\tau_c \propto \phi$  and again Fig. 7 confirms this dependence on  $\phi$  at high volume fractions.

Finally, comparing NNLs and LLs results in Fig. 7, it is apparent that using NNLs the time per HE collision is significantly smaller than in the case where only LLs are used. Moreover, it is far less density-dependent. This stems from the fact that in Eq. (72)  $N_{pc}^A$  and  $N_{pc}^B$  using NNLs are much smaller than the same quantities using LLs (see also discussion in the next section).

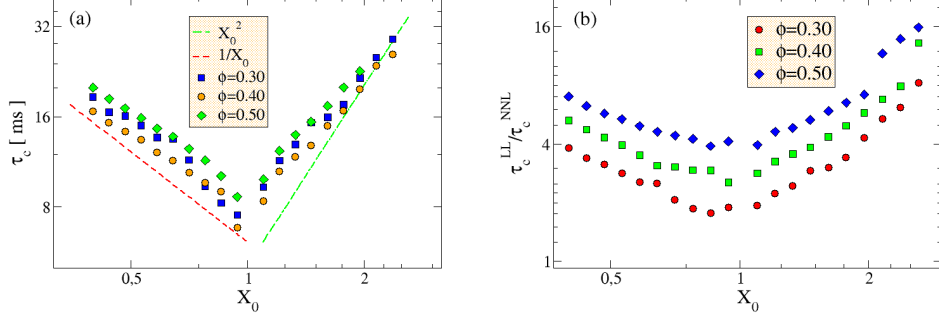


Figure 8: Dependence of average collision time  $\tau_c$  on elongation  $X_0$ . (a)  $\tau_c$  at  $\phi = 0.30, 0.40$  and  $0.50$  using only LLs. In this case the average collision time is linked to the number of particles in the first neighbour shell and grows with elongation. The dashed lines are guides to the eyes showing the  $1/X_0$  and  $X_0^2$  behaviors (see text). (b) Comparison of performance results with and without the use of NNLs plotting  $\tau_c^{LL}/\tau_c^{NNL}$  versus the aspect ratio  $X_0$  for 3 different volume fractions  $\phi = 0.30, 0.40, 0.50$ .  $\tau_c^{NNL}$  is the collision time employing NNLs and  $\tau_c^{LL}$  is the collision time using only LLs. The use of LLs severely degrades the performance of the algorithm for elongated HEs.

#### 4.3. Dependence of algorithm speed on elongation

In the following we will show results for simulations performed at volume fractions  $\phi = 0.30, 0.40, 0.50$ . First, the case where only LLs are employed is considered. Besides the expected decrease of  $\tau_c$  upon increasing the density, a marked increase of  $\tau_c$  upon increasing/decreasing the elongation with respect to the hard-sphere case  $X_0 = 1$  is clearly apparent in Fig. 8(a). The explanation for such behavior is straightforward: as noticed in [26], the number of HEs in a LL cell increases as  $X_0^2$  ( $1/X_0$ ) for the prolate (oblate) case, and in turns  $\tau_c$  increases as  $X_0^2$  ( $1/X_0$ ) at big (small) elongations. In contrast, using NNLs  $\tau_c$  is quite  $X_0$ -independent, increasing significantly the performance at big and small elongations with respect to LLs (see Fig. 8(b)). In this case the number of neighbors is independent from  $\phi$  on changing the elongation and the computational efficiency depends only on NR method efficiency, that is quite  $X_0$ -independent. On passing we note that at very high elongation ( $X_0 \gtrsim 5$ ), the NR method requires more steps to converge and the exact number of steps depends on the initial guess. Nevertheless, we checked that, at least up to  $X_0 = 10$ , the algorithm's performance does not change (i.e.  $\tau_c$  remains  $X_0$ -independent) if NNLs are used.

#### 4.4. Optimizing the parameter $\epsilon_d$

The main parameter entering in the collision prediction algorithm is  $\epsilon_d$  (sec. 2.4.2) and it is important to establish the optimal value in terms of efficiency. As already discussed in 2.4.2, too large a value for this parameter may result in a failure of EDMD due to overlaps of HEs. Hence, the best choice for  $\epsilon_d$  is the largest value not generating HEs overlaps. Fig. 9 shows that  $\tau_c$  monotonically decreases as  $\epsilon_d$  is decreased, and that  $\phi$ -dependence is rather weak. In practice, a value like  $\epsilon_d = 10^{-4} - 10^{-5}$  is usually a good, reliable and safe choice for most situations.

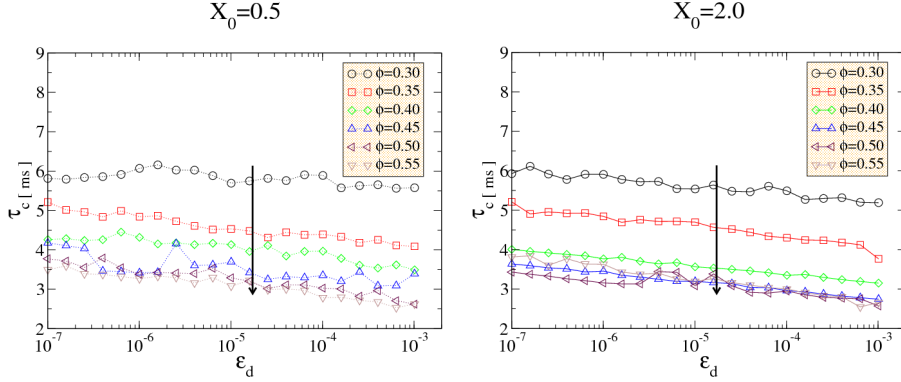


Figure 9: Dependence of  $\tau_c$  on  $\epsilon_d$  (using NNLs) for two different elongations ( $X_0 = 0.5, 2.0$ ) at various values of  $\phi$  ranging from 0.30 to 0.55. Arrows indicate decreasing volume fractions.

#### 4.5. Optimizing neighbour lists

Changing the parameter  $\Delta_{NNL}$ , i.e. changing the size of the bounding boxes, computational times vary non-monotonically. If  $\Delta_{NNL}$  tends to 0, the number of neighbors  $N_{pc}^A$ ,  $N_{pc}^B$  in Eq.(72) tends to 0 accordingly, but NNLs updates tend to be infinitely frequent, because escape time of HE from their bounding boxes tends to 0. Therefore  $\tau_c$  diverges to infinity, if  $\Delta_{NNL} \rightarrow 0$ . If  $\Delta_{NNL} \rightarrow \infty$  (assuming one has a system with infinite particles), escape times tends to 0 and time intervals between two successive NNLs rebuilds diverge, but  $N_{pc}^A \rightarrow \infty$ ,  $N_{pc}^B \rightarrow \infty$  in Eq.(72), so that again  $\tau_c \rightarrow \infty$ . As a result, there must be a minimum of  $\tau_c$  as a function of  $\Delta_{NNL}$ . The value of  $\Delta_{NNL}$ , which minimizes  $\tau_c$  is the optimal value for this parameter and it will be labeled  $\Delta_{NNL}^*$ . Fig. 10 shows for two volume fractions that  $\tau_c(\Delta_{NNL})$  exhibits a clear minimum.  $\Delta_{NNL}^*$  depends much more on the volume fraction  $\phi$  than on the elongation  $X_0$  and this is clear from Fig. 11, where  $\Delta_{NNL}^*$  is plotted as a function of  $\phi$  for different  $X_0$ . The latter result is due to the fact that in Eq. (72)  $\Delta t_{coll}$ , through  $N_{pc}^A$ ,  $N_{pc}^B$ , strongly depends on  $\phi$ , while both  $\Delta t_{coll}$  and  $\Delta t_u$  weakly depend on  $X_0$ .

Fig. 11 provides a simple estimate of  $\Delta_{NNL}^*$ , that ranges from 0.1 to 0.8 if  $0.5 < X_0 < 2.0$  and  $0.25 < \phi < 0.55$ , upon changing volume fraction and elongation.

### 5. Simulating Superquadrics.

In the Sections 3 and 4 we have shown in detail how to apply the algorithm illustrated in Sec. 2 to simulate HEs and its performance in this particular case. The algorithm proposed by Donev et al. [25, 26] has been generalized to arbitrary convex shapes [28, 29, 30, 31], analogously we consider here the simulation of SQs, that are a possible generalization of HEs studied in the previous section, skipping anyway some details that will be supplied in a future publication.

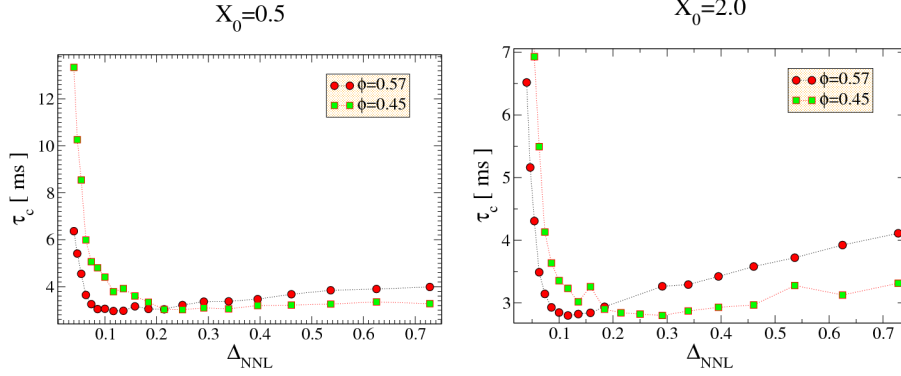


Figure 10: The average collision time depends non-monotonically upon  $\Delta_{NNL}$ , showing a clear minimum for a certain  $\Delta_{NNL}^*$  value (see text). In this figure such a dependence is shown for 4 particular state points ( $X_0 = 0.5$  at  $\phi = 0.45, 0.57$  and  $X_0 = 2.0$  at  $\phi = 0.45, 0.57$ ).

### 5.1. Geometry and Simulation Details

The surface of an HE has been defined through Eq. (42), that with a suitable choice of the reference system axes (principal axes) takes the form:

$$\left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 + \left(\frac{z}{c}\right)^2 = 1 \quad (76)$$

A straightforward generalization of the latter equation leads to the definition of particles, called SQs, whose surface is the following:

$$f(x, y, z) = \left|\frac{x}{a}\right|^n + \left|\frac{y}{b}\right|^m + \left|\frac{z}{c}\right|^p - 1 = 0 \quad (77)$$

where the parameters  $n, m, p$  are real numbers and  $a, b, c$  are the SQ semi-axes. A monodisperse system of  $N = 512$  SQs has been simulated with  $m = n = 2$  and with two equal semi-axes, i.e.  $b = c$ .

Such SQs can be characterized by the elongation  $X_0 = a/b$  and by the parameter  $p$ , that determines the sharpness of the edges (see Fig. 12). SQs of elongation  $X_0 < 1$  are called “oblate”, while SQs of elongation  $X_0 > 1$  are called “prolate”, as for the HEs. Elongations  $X_0 = 0.5$  and  $X_0 = 2.0$  have been studied. Nicely for  $X_0 = 0.5$  and  $p = 8$  the SQ has a pillow-like shape, while for  $X_0 = 2.0$  and  $p = 8$  it has a boat-like shape, as shown in Fig. 12. The system of prolate and oblate SQs has been simulated in a cubic box of volume  $V$  with periodic boundary conditions at the volume fraction  $\phi = 0.256$  and at various values of  $p$ . The length of the smallest semi-axes is chosen as 0.8 times the unit of distance, the mass of the SQ as unit of mass ( $m = 1$ ) and the moment of inertia is spherically symmetric, i.e.  $I_x = I_y = I_z = 1$ , as in the case of HEs.

The code for simulating SQs can be straightforwardly derived from the HE code, indeed for implementing the EDMD of SQs it suffices to:

- Evaluate the Jacobians in Eqs. (9) ( or Eqs. (23) ) and (31) using Eq. (77).

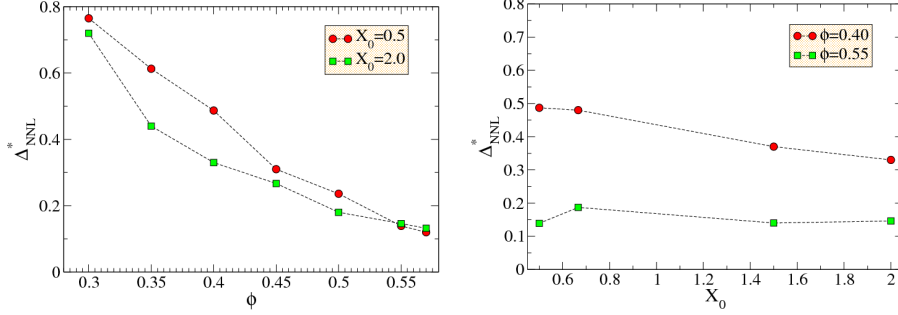


Figure 11: In this figure the dependance of  $\Delta_{NNL}^*$  on  $\phi$  for  $X_0 = 0.5, 2.0$  (left panel) and the dependance of  $\Delta_{NNL}^*$  on  $X_0$  for  $\phi = 0.40, 0.55$  (right panel) is shown.  $\Delta_{NNL}^*$  decreases significantly with increasing  $\phi$  (left panel) but it has a weak dependence on  $X_0$  (right panel).

- Calculate the Jacobians pertaining to the collision of a SQ with a plane to make use of NNL (see Sec. 3.3).
- Adapt the steepest descent method described in Sec. 2.3.2.
- Adapt the guess of the distance shown in Sec. 3.2.

### 5.2. Performance results

Fig. 13 shows the average collision time  $\tau_c$  for the system of SQs using either LLs or NNLs. With regard to the dependence of  $\tau_c$  on  $\phi$ ,  $X_0$ ,  $\epsilon_d$  and  $\Delta_{NNL}$  We do not expect a significantly different behavior from what has been observed for HEs. Hence here we focus on the dependence of the average collision time on the parameter  $p$ .

For the LLs case  $p$  ranges from 2 to 8, while enabling NNLs it ranges from 2 to 4 only. In the case of LLs for values of  $p > 8$  the NR for calculating the distance between SQs starts having convergence problems, while in the case of NNLs the NR for calculating the distance between a SQ and a plane starts having convergence issues. In the present implementation of the algorithm for locating the collision time between SQs and between a SQ and a plane the greater the  $p$  the slower will be the NR method to reach convergence (up to a maximum value above which the NR method starts failing) and the greater will be the number of steps needed to locate the contact point (see Sec. 2.4.2).

About the performance comparison between HEs and SQs from Fig. 13 it turns out that the SQs simulations (see  $p = 2$ , i.e. HEs) are at least 5-6 times slower than the HEs ones and this is mainly due to the more time consuming calculation of the Jacobians in Eqs (9), (23) and (31). Finally, as discussed above it is apparent from Fig. 13 the increase of  $\tau_c$  at high  $p$  in all cases.

## 6. Possible applications of the present algorithm

The algorithm illustrated in this paper has been already applied in several fields, like biophysics, soft-matter and condensed-matter physics. First of all, this new algorithm was employed to investigate the dynamics of HEs [43], finding, for the first time, evidence

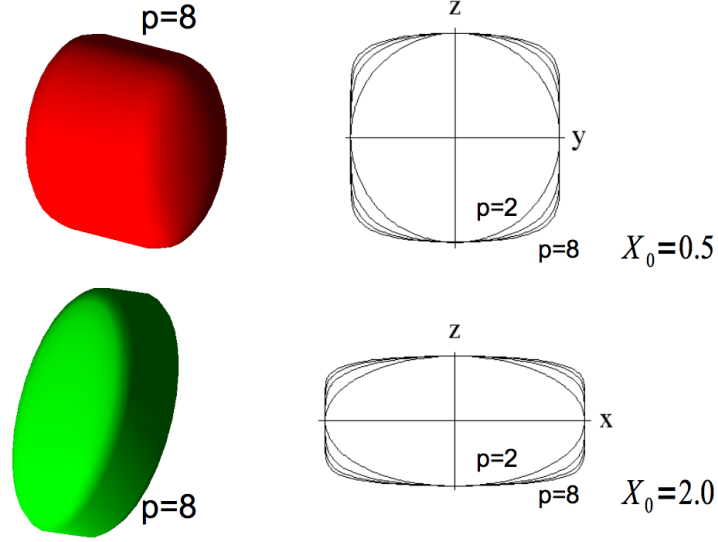


Figure 12: Shape of the superquadrics used in the simulations. Top: on the left a prolate SQ with  $p = 8$  and  $X_0 = 0.5$  is represented in 3D. On the right the plots of the contour resulting from the intersection of the SQ (for several  $p$  ranging from 2 to 8) with the plane  $x = 0$  are shown. Bottom: on the left an oblate SQ with  $p = 8$  and  $X_0 = 2.0$  is represented in 3D. On the right the plots of the contour resulting from the intersection of the SQ (for several  $p$  ranging from 2 to 8) with the plane  $y = 0$  are shown.

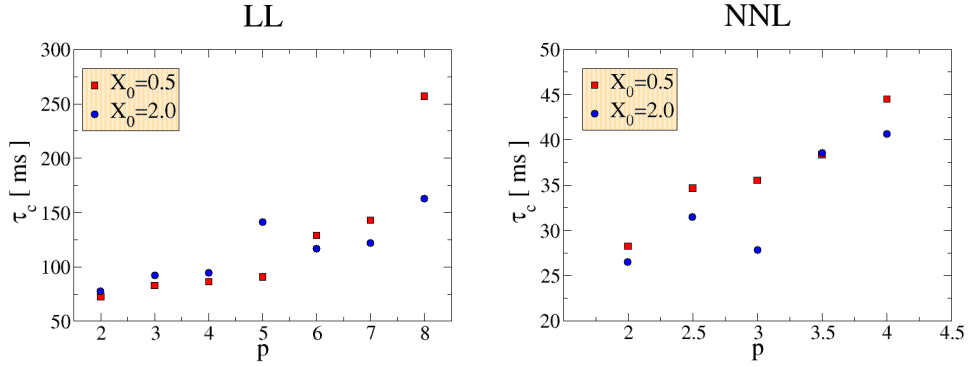


Figure 13: Average collision time  $\tau_c$  for simulations of  $N = 512$  SQs at  $\phi = 0.256$  for elongations  $X_0 = 0.5$  and  $X_0 = 2.0$  using only LL (left panel) and also using NNL (right panel).

of a pre-nematic order-driven glass transition. The fact that this algorithm can simulate hard objects of convex arbitrary shape offers new intriguing possibilities for investigating the changes in the phase diagram of HEs on changing particles shape. For example, concerning the pre-nematic order driven glass transition observed in [43], it is stimulating to think about the possibility to completely inhibit the nematic transition with a different choice of the HRB shape and/or introducing shape polydispersity. Work in this direction is under way. Changing the HRB shape may result in a phase diagram, which is richer than the one of simple HEs, since new phases (cubatic, smectic, ecc.) may appear [2]. Moreover, changing the shape and introducing shape polydispersity may have effects on packing and on mixing/demixing of hard objects.

This algorithm has also been extended [44] to combinations of hard and localized attractive interactions, modeled as site-site square well potentials. In this way, the hard-body may be decorated with an arbitrary number of spherical patches (sticky spots) arranged in fixed site locations. The algorithm for the prediction of the collision time between "sticky spots" is an adaptation of the algorithm illustrated in the present paper for HEs (some details can be found in [44] but this topic will be discussed extensively in a forthcoming publication).

Spheres decorated with sticky spots have been used to simulate primitive models of water [44] and silica [45] and to study the kinetics of self-assembly of an idealized fluid model [48]. Hard ellipsoids with sticky spots have been employed to study a model of a chemical gel [46].

HEs decorated with attractive interacting sites may also be used to model complex hard molecules, retaining some degree of flexibility. In this approach, the loss in the details of the system is counterbalanced by its extreme flexibility and by the possibility of investigating time scales which are not accessible by standard methods (like "ab-initio" calculations or full-atom molecular dynamics simulations). There are several directions in which this methodology could be applied. A very promising field of interest is represented by biophysical systems. A particular problem that has received much attention over the last years is the IgG antibody-antigen interaction. An immunoglobulin (or antibody) is a "Y" shaped protein that is ubiquitous in most vertebrates. Its role is to bind to viruses or bacteria to facilitate their neutralization process. It is extremely interesting that a single object has the ability to bind to biological targets whose size can vary from much smaller to much larger than its own. The introduction of square well interactions [44] between hard-body particles can be used to generalize the Go model [59] and better account for steric effects between different proteins. Previously, primitive model of proteins had been developed to study biophysical problems. In the 4-beads model, described in [60], each amino acid is represented by a maximum of four beads. Three beads correspond to the amide  $N$ , the  $\alpha$ -carbon  $C$ , the carbonyl  $C'$  groups. The fourth bead models the amino-acid side-chain group of atoms, and it is placed at the centre of the nominal  $C_\beta$  atom. Exploiting the new algorithm proposed in the present paper, each amino-acid with its interacting (active) sites can be modeled as a set of spherical spots that form one unique rigid body. In biological simulations it is worth noting that the solvent can be also introduced explicitly in several ways:

- using a primitive model, e.g. the primitive model of water used in [44] to take into account the directionality of the hydrogen bonding.
- using a simplified solvent of hard spherical particles (this technique has been used

for studying the interaction of IgG and antigenes)

- adapting the method for performing brownian dynamics of square-well particles discussed in [38] to the present algorithm.

Finally, this algorithm may find applications in the field of computer science: animations and virtual reality may benefit from the physical accuracy and efficiency of the present method. In general, virtual motions of objects in 3D space require an accurate prediction of collisions and are fundamental for robotics, computer graphics, 3D computer games [61] and CAD applications. For example, because of its flexibility in shape, ellipsoids are often chosen as bounding volumes for robotic arms in collision detection [62, 63, 64].

## 7. Conclusions

In this paper a new efficient method for performing event-driven molecular dynamics simulations of non-spherical HRBs has been proposed. This new method is based on the traditional NR method for solving a set of non-linear equations. In particular, geometrical distance and contact point and time between two moving HRBs can be calculated through the NR method in a very efficient way. The method has been tested against a well established and studied system, the hard ellipsoids of revolution and also against a less common system like the SQs. Anyhow HEs and SQs are just particular cases for the present algorithm, since more general convex hard bodies can be simulated. Furthermore also non-convex hard bodies can be in principle simulated by the present algorithm, if all the possible solutions of Eqs. (9) (or Eqs. (23)) can be found, because in this case the actual distance can be simply obtained by Eq. (11).

In the specific case of uniaxial HEs the time per collision achieved at moderate and high densities for elongations  $X_0 = 0.5, 2.0$  is around 2.5 ms, a value which is comparable to the performance of the method described in [26]. For comparison, in our implementation the time per sphere collision is about 0.25 ms, i.e. simulating hard spheres is about an order of magnitude faster than simulating HEs, even for nearly spherical HEs (the same observation has been carried out in [26]).

To solve the set of equations to evaluate the contact point and time (see Eqs. (30)) by the NR method, a good initial guess (“bracketing”) of the solution has to be provided. It has been shown (see 2.4.2) that an initial guess for Eqs. (30) can be found by evaluating the geometrical distance between the two colliding HRBs and by making use of a simple overestimate of the rate of variation of the distance ( $\dot{d}_{max}$ ). It is worth stressing that using such an algorithm for bracketing the contact point and time, “grazing collisions” do not constitute a problem within machine accuracy. If  $\epsilon_d \lesssim 10^{-4}$  all collisions are correctly predicted and this choice of  $\epsilon_d$  does not depend on volume fraction or aspect ratio of simulated HEs. Also a new method to implement NNL based on oriented bounding boxes has been presented. This new NNL method is fast and flexible enough to be easily extended to more complex shapes than simple HEs and SQs.

The present algorithm can also be generalized, without losing numerical efficiency, to the case of attractive interactions modeled via discontinuous step-wise potentials, including the interesting case of site-site square well interaction, providing the possibility of modeling highly directional interactions between the rigid bodies. This possibility can



be further refined by transforming site-site interactions into a permanent link between objects, by simply changing the depth of square well interactions. In this way, the objects can be linked into flexible structures. Such flexibility couples well with the novel NNL implementation described.

## 8. Acknowledgments

The author acknowledges support from CASPUR and Cofin. The author warmly thanks Prof. F. Sciortino, Dr. G. Foffi, F. Romano and A. De Michele for their careful reading of the manuscript, Prof. P. Tartaglia for his ongoing support and Prof. G. Ciccotti for helpful discussions. The author also thanks Dr. A. Scala for useful discussions and for having taken part to the very early stages of this project.

## References

- [1] M. P. Allen, Liquid crystal systems, in: N. Attig, K. Binder, H. Grubmüller, K. Kremer (Eds.), *Computational Soft Matter: From Synthetic Polymers to Proteins*, Vol. 23, John von Neumann Institute for Computing, Jülich, 2004, pp. 289–320.
- [2] D. Frenkel, B. M. Mulder, The hard ellipsoid-of-revolution fluid, *Mol. Phys.* 55 (1985) 1171.
- [3] M. P. Allen, G. Evans, D. Frenkel, B. M. Mulder, Hard convex body fluids, *Adv. Chem. Phys.* 86 (1993) 1–166.
- [4] J. Talbot, D. Kivelson, M. P. Allen, G. T. Evans, D. Frenkel, Structure of the hard ellipsoid fluid, *J. Chem. Phys.* 92 (1990) 3048–3057.
- [5] M. P. Allen, M. A. Warren, Simulation of structure and dynamics near the isotropic-nematic transition, *Phys. Rev. Lett.* 78 (1997) 1291–1294.
- [6] C. G. Gray, K. E. Gubbins, *Theory of molecular fluids*, Vol. 1, Clarendon Press, 1984.
- [7] H. C. Andersen, D. Chandler, J. D. Weeks, Roles of repulsive and attractive forces in liquids: The equilibrium theory of classical fluids, *Adv. Chem. Phys.* 34 (1976) 105.
- [8] J. D. Weeks, D. Chandler, H. C. Andersen, Roles of repulsive forces in determining the equilibrium structure of simple liquids, *J. Chem. Phys.* 54 (1971) 5237–5247.
- [9] J. P. Hansen, I. R. McDonald, *Theory of Simple Liquids*, 2nd Edition, Academic Press, 1986.
- [10] J. Kolafa, I. Nezbeda, Monte carlo simulations on primitive models of water and methanol, *Mol. Phys.* 61 (1987) 161.
- [11] P. N. Pusey, *Liquids, Freezing and the Glass Transition*, Vol. 2, North-Holland, Amsterdam, 1991, Ch. 10, p. 763.
- [12] G. Foffi, W. Götze, F. Sciortino, P. Tartaglia, Th. Voigtmann, Mixing effects for the structural relaxation in binary hard-sphere liquids, *Phys. Rev. Lett.* 91 (2003) 085701.
- [13] G. Foffi, W. Götze, F. Sciortino, P. Tartaglia, Th. Voigtmann,  $\alpha$ -relaxation processes in binary hard-sphere mixtures, *Phys. Rev. E* 69 (2004) 011505.
- [14] L. Onsager, The effects of shape on the interaction of colloidal particles, *Ann. N. Y. Acad. Sci.* (1949) 627.
- [15] P. G. de Gennes, J. Prost, *The Physics of Liquid Crystals*, Oxford University Press, 1995.
- [16] G. J. Vroege, H. N. W. Lekkerkerker, Phase transitions in lyotropic colloidal and polymer liquid crystals, *Rep. Prog. Phys.* 55 (1992) 1241–1309.
- [17] J. D. Parsons, Nematic ordering in a system of rods, *Phys. Rev. A* 19 (1979) 1225.
- [18] S.-D. Lee, The onsager-type theory for nematic ordering of finite-length hard ellipsoids, *J. Chem. Phys.* 89 (1989) 7036.
- [19] A. Samborski, G. T. Evans, C. P. Mason, M. P. Allen, The isotropic to nematic liquid crystal transition for hard ellipsoids: An onsager-like theory and computer simulations, *Mol. Phys.* 81 (1994) 263–276.
- [20] B. Tijpto-Margo, G. T. Evans, The onsager theory of the isotropic-nematic liquid crystal transition: incorporation of the higher virial coefficients, *J. Chem. Phys.* 93 (1990) 4254.
- [21] D. Frenkel, B. M. Mulder, J. P. McTague, Phase diagram of a system of hard ellipsoids, *Phys. Rev. Lett.* 52 (1984) 287–290.

- [22] B. J. Alder, T. E. Wainwright, Phase transition for a hard sphere system, *Journal of Chemical Physics* 27 (1957) 1208–1209.
- [23] J. Vieillard-Baron, Phase transitions of the classical hard-ellipse system, *Journal of Chemical Physics* 56 (1972) 4729–4744.
- [24] J. Perram, M. Wertheim, Statistical mechanics of hard ellipsoids. i. overlap algorithm and the contact function, *J. Comp. Phys.* 58 (1985) 409–416.
- [25] A. Donev, S. Torquato, F. H. Stillinger, Neighbor list collision-driven molecular dynamics simulation for nonspherical hard particles. i. algorithmic details, *Journal of Computational Physics* 202 (2005) 737–764.
- [26] A. Donev, S. Torquato, F. H. Stillinger, Neighbor list collision-driven molecular dynamics simulation for nonspherical hard particles. ii. applications to ellipses and ellipsoids, *Journal of Computational Physics* 202 (2005) 765–793.
- [27] A. Donev, I. Cisse, D. Sachs, E. A. Variano, F. H. Stillinger, R. Connelly, S. Torquato, P. M. Chaikin, Improving the density of jammed disordered packings using ellipsoids, *Science* 303 (2004) 990.
- [28] A. Donev, Jammed packing of hard particles, Ph.D. thesis, Princeton University (2006).
- [29] A. Donev, J. Burton, F. H. Stillinger, S. Torquato, Tetratic order in the phase behavior of a hard-rectangle system, *PRB* 73 (2006) 054109.
- [30] Y. Jiao, F. H. Stillinger, S. Torquato, Superdisks and the role of symmetry, *PRL* 100 (2008) 245505.
- [31] Y. Jiao, F. H. Stillinger, S. Torquato, Optimal packings of superballs, *PRE* 79 (2009) 0141309.
- [32] D. Frenkel, B. Smit, *Understanding molecular simulation : from algorithms to applications*, 2nd Edition, Academic Press, 2002.
- [33] R. Blaak, D. Frenkel, B. M. Mulder, Do cylinders exhibit a cubatic phase?, *Journal of Chemical Physics* 100 (1999) 11652–11659.
- [34] J. A. C. Veerman, D. Frenkel, Phase behavior of disklike hard-core mesogens, *Phys. Rev. A* 45 (1992) 5632–5648.
- [35] M. Het, P. Siders, Monte carlo calculation of orientationally anisotropic pair distributions and energy transfer in a model monolayer, *J. Phys. Chem.* 94 (1990) 7280–7288.
- [36] D. C. Rapaport, *The Art of Molecular Dynamics Simulation*, Cambridge University Press, 2004.
- [37] B. Lubachevsky, How to simulate billiards and similar systems, *J. Comput. Phys.* 94 (1991) 255–283.
- [38] A. Scala, C. De Michele, Th. Voigtmann, Event-driven brownian dynamics for hard spheres, *J. Chem. Phys.* 126 (2007) 134109.
- [39] A. Donev, Asynchronous event-driven particle algorithms, *Simulation* 85 (2009) 229–242.
- [40] M. Allen, D. Frenkel, J. Talbot, Molecular dynamics simulation using hard particles, *Comput. Phys. Rep.* 9 (1989) 301–353.
- [41] M. P. Allen, Diffusion coefficient increases with density in hard ellipsoid liquid crystals, *Phys. Rev. Lett.* 65 (1990) 2881.
- [42] C. De Michele, A. Scala, R. Schilling, F. Sciortino, Molecular correlation functions for uniaxial ellipsoids in the isotropic state, *J. Chem. Phys.* 124 (2006) 104509.
- [43] C. De Michele, F. Sciortino, R. Schilling, Dynamics of uniaxial hard ellipsoids, *Phys. Rev. Lett.* 98 (2007) 265702.
- [44] C. De Michele, S. Gabrielli, P. Tartaglia, F. Sciortino, Dynamics in the presence of attractive patchy interactions, *J. Phys. Chem. B* 110 (2006) 8064.
- [45] C. De Michele, P. Tartaglia, F. Sciortino, Slow dynamics in a primitive tetrahedral network model, *J. Chem. Phys.* 125 (2006) 204710.
- [46] S. Corezzi, C. De Michele, E. Zaccarelli, D. Fioretto, F. Sciortino, A molecular dynamics study of chemical gelation in a patchy particle model, *Soft Matter* 4 (2008) 1173–1177.
- [47] S. Corezzi, C. De Michele, E. Zaccarelli, P. Tartaglia, F. Sciortino, Connecting irreversible to reversible aggregation: Time and temperature, *The Journal of Physical Chemistry B* 113 (5) (2009) 1233–1236.
- [48] F. Sciortino, J. Douglas, C. De Michele, Growth of equilibrium polymers under non-equilibrium conditions, *J. Phys.: Condens. Matter* 20 (2008) 155101.
- [49] L. Landau, E. Lifshitz, *Course of theoretical physics, Vol. 1, Mechanics*, Pergamon Press, 1960.
- [50] E. T. Whittaker, *A treatise on the analytical dynamics of particles and rigid bodies*, 4th Edition, Cambridge University Press, 1947.
- [51] R. van Zon, J. Schofield, Numerical implementation of the exact dynamics of free rigid bodies, *J. Chem. Phys.* 128 (2008) 154119.
- [52] L. H. de la Pena, R. van Zon, J. Schofield, S. B. Opps, Discontinuous molecular dynamics for semi-flexible and rigid bodies, *J. Chem. Phys.* 126 (2007) 074105.

- [53] H. Goldstein, Classical Mechanics, 2nd Edition, Addison Wesley, 1980.
- [54] A. E. Taylor, R. M. Mann, Advanced Calculus, John Wiley and Sons, 1983.
- [55] W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling, Numerical Recipes in Fortran, Cambridge University Press, 1999.
- [56] P. W. Cleary, N. Stokesand, J. Hurley, Efficient collision detection for three dimensional super-ellipsoid particles, in: Proc. 8th International Computational Techniques and Applications Conference, World Scientific, 1997, pp. 1–7.
- [57] C. Ericson, Real-Time Collision Detection, Morgan Kaufmann, 2005.
- [58] M. P. Allen, D. J. Tildesley, Computer simulation of liquids, paperback 385pp Edition, Clarendon Press, 1989.
- [59] N. Go, Theoretical studies of protein folding, Annu Rev Biophys Bioeng. 12 (1983) 183–210.
- [60] L. Cruz, S. Yun, S. V. Buldyrev, G. Bitan, D. B. Teplow, H. E. Stanley, Discrete molecular dynamics simulations of peptide aggregation, PNAS 101 (2004) 17345–17350.
- [61] D. H. Eberly, 3D Game Engine Design, Academic Press, 2001.
- [62] M. Ju, J. Liu, S. Shiang, Y. Chien, K. Hwang, W. Lee, A novel collision detection method based on enclosed ellipsoid, in: Proceedings of 2001 IEEE Conference on Robotics and Automation, 2001, pp. 21–26.
- [63] E. Rimon, S. Boyd, Obstacle collision detection using best ellipsoid fit, Journal of Intelligent and Robotic Systems 18 (1997) 105–126.
- [64] C. J. Wu, On the representation and collision detection of robots, Journal of Intelligent and Robotic Systems 16 (1996) 151–168.